國立臺灣大學電機資訊學院資訊網路與多媒體研究所
碩士論文

Graduate Institute of Networking and Multimedia
College of Electrical Engineering & Computer Science
National Taiwan University
Master Thesis

用於伴奏分離的輕量化深度學習模型

Lightweight Deep-Learning Models for Accompaniment Separation

王俊翔

Chun-Hsiang Wang

指導教授：張智星　博士

Advisor: Jyh-Shing Roger Jang, Ph.D

中 華 民 國 一百一十二 年 八 月

Aug, 2023

doi:10.6342/NTU202303861

doi:10.6342/NTU202303861

國立臺灣大學碩士學位論文
口試委員會審定書

用於伴奏分離的輕量化深度學習模型

Lightweight Deep-Learning Models for Accompaniment Separation

本論文係王俊翔君（學號 R10944023）在國立臺灣大學資訊網路與多媒體研究所完成之碩士學位論文，於民國一百一十二年七月十日承下列考試委員審查通過及口試及格，特此證明。

口試委員：張智星（簽名）

（指導教授）

楊奕軒　　曹昱

所　長：鄭卜壬

doi:10.6342/NTU202303861

誌謝

首先，要感謝我的指導教授張智星老師。在這兩年的碩班生涯當中，是他帶我進入音樂相關的研究領域，並且總是在我最需要幫助時為我指引方向，尤其在鄰近口試時老師的細心指導讓我成長許多，沒有老師的協助我絕對無法順利完成口試。另外要特別感謝王崇喆學長及葉子雋學長這兩年的大力協助，不論是在參與的計畫當中或是論文口試都提供我許多的思考方向，讓我以不同角度看待同一件事情，使我拓展視野、增加研究的深度及廣度。感謝與我一同待在 MIRLAB 的同學們，與他們的相處讓我受益良多，讓我對研究不再感到枯燥乏味，尤其是與我攜手進行相同計畫的李學翰和總是不吝提供建議的王鈞右。感謝我的好友們，尤其是譚庚豪及簡睿閎同學，是他們伴我在研究之餘繼續享受人生，豐富了我的生活點滴。最後要感謝我的家人，他們總是時時刻刻關心著我的日常生活，讓我衣食無憂，專心的進行研究。在未來的日子當中，我絕對不會忘記這兩年的點點滴滴，這兩年的經驗也將助我突破往後人生中的種種挑戰。

doi:10.6342/NTU202303861

誌謝

doi:10.6342/NTU202303861

用於伴奏分離的輕量化深度學習模型

摘要

音樂聲源分離旨在分離出一首歌曲當中的不同音軌，本篇研究著重於對伴奏軌的即時分離。過去的音樂聲源分離模型傾向於提升分離品質，但這使得模型的大小和延遲時間增加，難以在邊緣裝置上進行運算。此外，大多數方法在降低輸入秒數時會明顯降低分離品質。本論文改進 Sony 早期提出的輕量化模型 MMDenseNet，希望在分離品質、延遲時間及空間資源三者間達成平衡。儘管 MMDenseNet 的參數量很低，但分離品質不理想，且在低延遲時情況下表現不佳。因此，本研究提出了三個改進方向，分別為訓練目標調整、模型架構調整、以及訓練及測試方法改進，試圖在維持空間資源表現的情形下改善模型分離品質與延遲時間。我們使用 MUSDB18 資料集進行訓練及測試，並使用 SDR 作為分離品質評估指標，計算延遲時間作為延遲評估指標，使用參數量作為空間資源評估指標。根據實驗結果，調整模型的訓練目標能夠在維持空間資源與延遲時間下提高分離品質 (median SDR 從 11.162 提升至 13.951)。此外，提出的多種自注意力架構使 MMDenseNet 在稍微增加空間資源及延遲時間的情況下提升分離品質 (median SDR 從 13.951 提升至 15.011)。最後，我們提出的漸進式訓練及測試方法使得模型在低延遲下能夠保持良好的分離品質 (延遲時間 1.19 秒時，RTF 為 0.4031、median SDR 由 13.951 提升至 14.394)。

關鍵字： 伴奏分離、MMDenseNet、自注意力機制、漸進式訓練、U-Net

doi:10.6342/NTU202303861

doi:10.6342/NTU202303861

Lightweight Deep-Learning Models for Accompaniment Separation

Abstract

Music source separation aims to separate different tracks within a song, and this study focuses on the real-time separation of the accompaniment track. Previous music source separation models have prioritized improving separation quality, but this has resulted in increased model size and latency, making it challenging to perform computations on edge devices. Additionally, most methods exhibit a significant decrease in separation quality when reducing the input duration. This paper improves the lightweight model MMDenseNet proposed by Sony in the early stages, aiming to achieve a balance between separation quality, latency, and spatial resources. Although MMDenseNet has a low number of parameters, its separation quality is not ideal and it performs poorly in low-latency scenarios. Therefore, this research proposes three improvement directions, including adjustments to the training objectives, model architecture, as well as improvements in the training and testing methods, aiming to enhance the separation quality and latency while maintaining the parameter count. We utilize the MUSDB18 dataset for training and testing, using SDR as the evaluation metric for separation quality, measuring latency as the delay evaluation metric, and considering parameter count as an indicator of spatial resources. Based on the experimental results, adjusting the model's training objectives improves the separation quality while maintaining spatial resources and latency (median SDR improves from 11.162 to 13.951). Furthermore, the proposed various self-attention architectures enable MMDenseNet to enhance the separation quality with only a slight increase in spatial resources and latency (median SDR improves from 13.951 to 15.011). Finally, the progressive training and testing methods proposed in this study allow the model to maintain good separation quality at low

doi:10.6342/NTU202303861

latency (at a delay of 1.19 seconds, RTF is 0.4031, and median SDR improves from 13.951 to 14.394).

Keywords: accompaniment separation, MMDenseNet, self-attention mechanism, progressive training、U-Net

doi:10.6342/NTU202303861

目錄

doi:10.6342/NTU202303861

doi:10.6342/NTU202303861

doi:10.6342/NTU202303861

doi:10.6342/NTU202303861

圖目錄

頁次

doi:10.6342/NTU202303861

doi:10.6342/NTU202303861

表目錄

頁次

doi:10.6342/NTU202303861

doi:10.6342/NTU202303861

第一章　緒論

1.1　研究主題簡介

音樂聲源分離 (music source separation, MSS) 旨在將音樂分離為不同的樂器聲源，如鼓聲、歌聲、伴奏等。在深度學習模型的快速發展下，分離效果相較傳統方法有顯著的提升。然而，在追求分離效果的同時，往往伴隨著參數量龐大、分離時間過長等問題導致模型難以套用至嵌入式裝置中。相關研究的深度學習模型參數級距相差甚大，Sony 發表的 MMDenseNet [1] 為輕量化架構，但分離效果普通、Meta 提出的 HT Demucs [2] 模型效果突出，參數量卻是 MMDenseNet [1] 的百倍之多。

音樂聲源分離技術可以被廣泛應用至不同場景像是卡拉 OK、歌曲混音等，也時常被使用在其他研究主題的前置處理階段，像是歌詞對位、人聲轉換都需要有出色的人聲分離技術輔助以達成更好的效果，本論文主要探討音樂聲源分離的其中一個分支 - 伴奏分離，並改進輕量化模型 MMDenseNet [1]，在參數量精簡的前提下優化分離效果。

1.2　相關研究簡介

本章主要探討目前深度學習模型應用於音樂聲源分離的技術，其中大部分的模型都是使用 U-Net 為基礎架構進行延伸。此外，依照模型輸入大致可以分為三類：波形導向 (waveform based)、時頻譜導向 (spectrogram based) 及混合導向。

1.2.1　波形導向方法

使用波形導向方法的模型直接將音訊視為輸入，並未做額外的前置處理。此類方法的先驅為 Wave-U-Net [3]，Wave-U-Net[3] 由多層卷積神經網路 (CNN) 組成編碼器及解碼器，此後奠定了由 U-Net [4] 為主體的架構；為了使模型觀察更

doi:10.6342/NTU202303861

長時間的資訊，Demucs [5] 在編碼器及解碼器之中添加了雙向長短期記憶網路 (Bi-LSTM) [6]。Conv-Tasnet [7] 在語者分離領域獲得成功，其主要結構為 dilated residual convolution，在調整了 CNN 的設定後，套用至音樂聲源分離也展現不錯的效果。

1.2.2 時頻譜導向方法

使用時頻譜做為音樂聲源分離的輸入為最常見的方式，將輸入音源經過短時距傅立葉轉換 (STFT) 後可以對頻率做更進一步的分析，常見的方式是取其強度 (magnitude) 做為輸入，如本論文的基準模型 MMDenseNet [1] 也是採用此類方法；近期的做法不再侷限於預測時頻譜強度，相位 (phase) 的資訊也被考慮進去，在 PhaseNet [8] 當中，相位的預測被轉化為分類問題，把不同的角度切分為多個不同區塊藉此來做相位的估計；最近的做法則以複數頻譜圖 (complex spectrogram) 做為輸入，並預測複數理想比例遮罩 (complex ideal ratio mask, cIRM)，band-split RNN [9] 為此方法中分離表現最為突出的模型。

1.2.3 混合導向方法

混合導向方法將時域與頻域的資訊皆納入考量中，KUIELab-MDX-NET [10] 使用模型融合 (model blending) 方法對時域與頻域的輸出做加權平均，Hybrid Demucs [11] 及 HT Demucs [2] 則使用了 bi-U-Net 的模型架構將波形及時頻譜同時當作模型輸入及預測輸出，其中 HT demucs [2] 加入了除 MUSDB18 [12] 資料集外的訓練資料後可以達到現今最高 SDR (signal-to-distortion ratio)。

1.3 本論文方法簡介及創新之處

在本論文中，我們的目標是在伴奏分離任務中，達成高分離品質、低延遲時間、低空間資源三者的平衡。分離品質方面，我們以 median SDR 值大於 14 為標準，並希望盡量提高。延遲時間方面，我們希望能夠降低至接近一秒，並且 RTF 小於 1，如此才能夠進行即時分離。空間資源方面，我們希望模型的參數量在 1M 以下。要達成此目標，可以從分離品質高的模型逐步減少參數量，抑或是改善分離品質較低的輕量化模型，本論文選擇後者來進行實驗，其主要原因為目前分離品質較高的模型距離低延遲時間與低空間資源過於遙遠。

doi:10.6342/NTU202303861

本論文使用 MMDenseNet [1] 為基準模型，並引用多種方法改進分離品質，主要方法如下：

- 調整訓練目標。以强度遮罩 (magnitude mask) 與複數理想比例遮罩 (complex Ideal Ratio Mask, cIRM) 取代直接預測時頻譜强度大小，在參數量幾乎不變的情況提升分離品質。
- 調整模型架構。提出三種不同的自注意力機制架構，使模型考慮全局特徵，提高分離品質並盡量維持模型參數量及延遲時間。
- 減少延遲時間。基於本論文提出的模型架構，提出漸進式測試與漸進式訓練方法，試圖在縮短歌曲輸入片段時維持分離品質。

其中我們提出的區塊式軸向自注意力機制、New cIRM MMDenseNet 能使分離品質有顯著的提升，漸進式訓練和測試方法能有效減少延遲降低對分離表現的影響，為本論文的亮點及主要貢獻。

1.4 章節概述

本論文總共分為六章節，章節概述如下:

- 諸論: 簡述研究主題、研究方法及相關研究。
- 文獻探討: 說明本論文於研究方法中提及之相關研究。
- 研究方法: 介紹基於目前方法的改進方式。
- 資料集與實驗設定: 介紹使用之資料集與實驗設定。
- 實驗結果與討論: 分析並探討實驗結果。
- 總結: 簡述結論及未來展望。

doi:10.6342/NTU202303861

doi:10.6342/NTU202303861

第二章　文獻探討

本章主要針對本論文中參考到的技術細節做完整的說明。第一部分會詳細介紹我們使用的基準模型 MMDenseNet [1]，第二部分將探討音樂聲源分離模型的不同訓練目標，第三部分將簡單介紹自注意力機制，最後則會介紹 band-split RNN [9] 架構。本章所提到的技術皆為前人幾經驗證所提出，本論文受到啓發後將這些技術應用於改進基準模型當中。

2.1　MMDenseNet

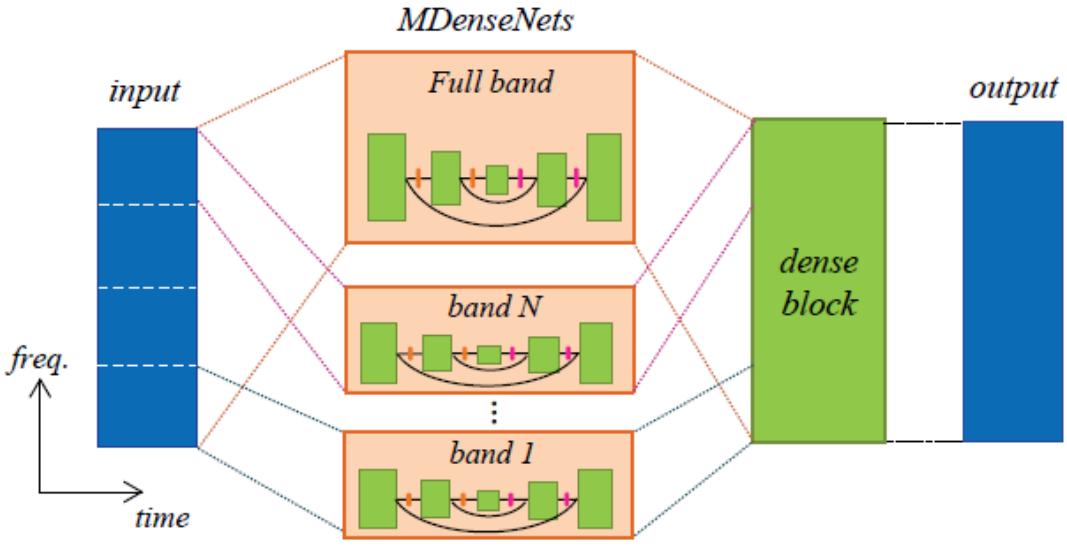


圖 2.1: 本論文使用的基準模型 MMDenseNet 架構 [1]

MMDenseNet [1] 於 2017 年由 Sony 所提出，其特點在於參數量精簡、模型訓練快速，並且分離表現優於 2016 年舉辦之 Signal Separation Evaluation Campaign (SiSEC 2016) [13] 的所有方法。在這一節當中，第一部份我們會介紹模型基礎架構 DenseNet [14]，第二部份會介紹作者如何延伸 DenseNet [14] 至 MDenseNet；最

doi:10.6342/NTU202303861

後，我們詳細說明 MMDenseNet 的完整架構。

2.1.1 DenseNet

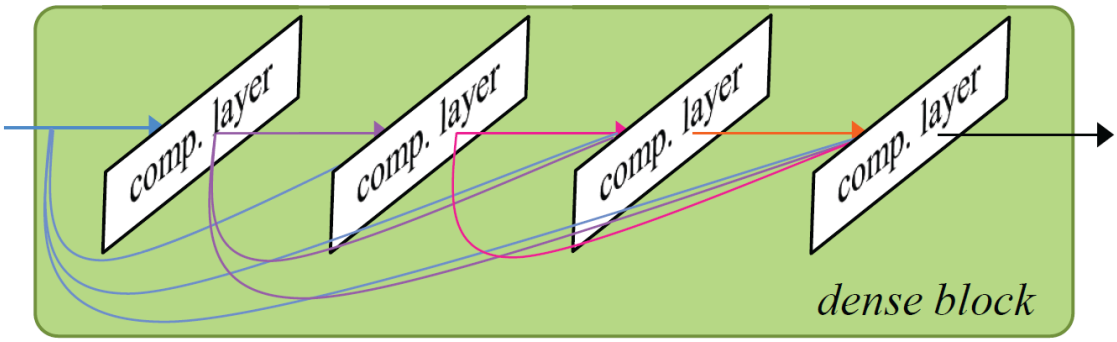


圖 2.2: Dense block 架構圖 [14]。圖中 comp. layer 為 composite layer，為非線性轉換的操作。

DenseNet 為卷積神經網路 (CNN) 中著名的架構。在進行深度學習時，加入多層的神經網路有助於提昇模型的準確率，然而當疊加的層數越來越多時便會使訓練難度增加。為了緩解問題，K. He [15] 等人提出了殘差神經網路 (ResNet)，其特點在於利用跳躍連接 (skip connection) 的技巧使模型在越來越深的同時依舊能看到較早的資訊以提昇訓練效果，每層的連接方式如下：

$$x_l = H_l(x_{l-1}) + x_{l-1} \tag{2.1}$$

其中 x_0 為輸入，$H_l(\cdot)$ 為非線性轉換 (可以是一連串的函式組合如歸一化、激勵函數、池化層、卷積層等)。基於此基礎下，G.Huang [14] 等人提出了 DenseNet。DenseNet 延伸了 ResNet [15] 的想法，將跳躍連接廣泛使用至各層當中，例如在 ResNet 中，第 l 層僅會與 $l-1$ 層使用跳躍連接並相加，而在 DenseNet 的 dense block (圖2.2) 當中則會與前面的 $l-1$ 相互連接並結合，其公式如下：

$$x_l = H_l([x_0, x_1, ..., x_{l-1}]) \tag{2.2}$$

2.1.2 Multi-Scale DenseNet (MDenseNet)

MDenseNet (圖 2.3) 主要分為編碼器及解碼器兩部份，中間以一個 dense block 作為連接，我們稱其為 bottleneck layer，且編碼器與解碼器的各區塊相互對應。

doi:10.6342/NTU202303861

編碼器的組成包含了 dense block 和 down-sampling layer，每個 dense block 後銜接一個 down-sampling layer，在各個 down-sampling layer 當中則使用了一個 1 x 1 的卷積層和一個 kernel size 為 2 x 2 的平均池化層 (average pooling layer)。在下採樣的過程當中，特徵圖 (feature map) 的分辨率逐漸降低，因此卷積層越接近 bottleneck 越能觀察到宏觀的視野。

解碼器包含了 dense block 及 up-sampling layer。up-sampling layer 使用了轉置卷積 (transposed convolution)，其 filter size 跟編碼器中池化層的 kernel size 相等，我們交替使用 up-sampling layer 及 dense block，如此可以逐步恢復更高分辨率的特徵圖，最終的輸出會與輸入大小相等。除此之外，作者引入了 U-Net [4] 中常見的跳躍連接，即在編碼器和解碼器相同尺度下的 dense blocks 進行連接，這樣的設計有助於 down-sampling layer 直接傳輸尚未壓縮的特徵圖給解碼器使之觀測不同尺度的資訊來進行預測。

最後，我們可以發現此架構為全卷積架構，因此可以輸入任意長度的時頻譜，正好符合本論文的低延遲需求。

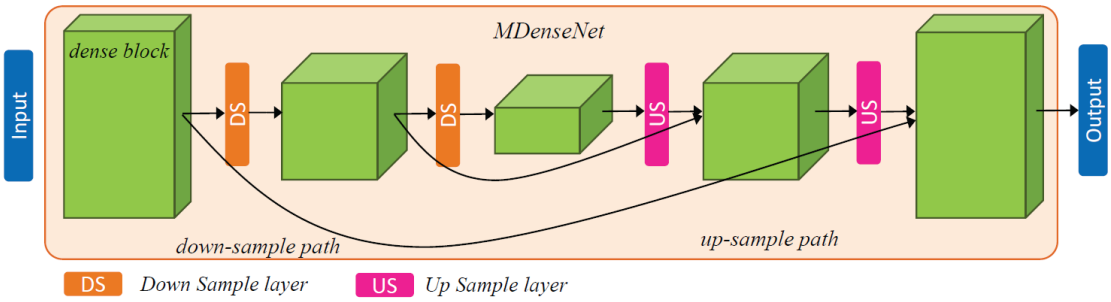


圖 2.3: MDenseNet 架構圖 [1]

2.1.3 Multi-band MDenseNet (MMDenseNet)

在 2.1.2 中，輸入為一個時頻譜，預測結果為分離過後的聲源，其中的卷積核 (convolution kernel) 共享整個時頻譜；然而這樣的設計方式在以影像為基礎的輸入上較為合理，原因是影像中出現的型態特徵較容易出現在任意位置當中。在音訊的輸入中，不同頻帶通常擁有不同的形態特徵，以全局來概括顯然不太合理，將不同的卷積核套用至不同頻帶上為較合適的選擇，事實上在語音辨識領域此理論已被印證。

MMDenseNet [1] 將上述理論整合並融入進 MDenseNet，觀察其模型架構 (圖 2.1) 可以發現其主要的模型骨架由數個不同大小的 MDenseNet 所組成，每一個

doi:10.6342/NTU202303861

MDenseNet 分別對映不同頻帶的輸入，使其專注於局部特徵的擷取，此外，作者還另外構建了一個用於完整頻帶輸入的 MDenseNet，以此來捕捉全局特徵。這些 MDenseNets 的輸出隨後被結合，最後經過 dense block 後預測出強度時頻譜。

2.2 音樂聲源分離訓練目標介紹

在這一章節當中，我們將介紹多種不同的音樂聲源分離模型訓練目標，通過前人的實驗，我們可以發現即便使用相同的模型架構，預測的準確度也會與模型最終的訓練目標有極大的關聯；本論文主要探討的範圍限縮於時頻譜的不同預測方法，在進入內容之前，我們須事先定義一些參數以便進行後續說明。

- $x, s \in \mathbb{R}_L$
  x: 時域中的混音訊號
  s: 時域中的聲源訊號，亦即分離目標之音訊
  L: 時間訊號的採樣點數量

- $X, S \in \mathbb{C}^{T \times F}$
  X, S: x, s 的短時距傅立葉轉換 (short-time Fourier transform, STFT)
  T: 轉換後的幀數
  F: 轉換後的頻帶數量

- $X_r, S_r \in \mathbb{R}^{T \times F}$: X, S 的實部

- $X_i, S_i \in \mathbb{R}^{T \times F}$: X, S 的虛部

- $|*| \in \mathbb{R}^{T \times F}$: 訊號經短時距傅立葉轉換後的強度值

- $\angle * \in [-\pi, \pi]^{T \times F}$: 訊號經短時距傅立葉轉換後的相位值

2.2.1 強度時頻譜預測

強度時頻譜預測為最直接的模型預測目標；在此方法中，模型學習一個直接由 $|X|$ 映射成 $|S|$ 的函式，等式如下：

$$|\hat{S}| = f(|X|) \tag{2.3}$$

doi:10.6342/NTU202303861

其中 $f(*)$ 可以是任意函式組合或神經網路。單憑預測的強度時頻譜 $|\hat{S}|$，我們無法還原由強度及相位組成的 STFT，於是通常會直接使用混音訊號的相位 STFT 當作預測，等式如下：

$$\hat{S} = |\hat{S}|e^{j\angle X} \tag{2.4}$$

最後再使用反短時距傅立葉轉換 (inverse short-time Fourier transform, iSTFT) 得到預測的分離訊號:

$$\hat{s} = iSTFT(\hat{S}) \tag{2.5}$$

2.2.2 強度遮罩預測

強度遮罩 (magnitude mask) 為常見的模型預測目標，此類模型學習一個遮罩 $|\hat{M}| \in \mathbb{R}_{\geq 0}^{T \times F}$ 與混音訊號的強度時頻譜進行逐元素相乘 (element-wise multiplication) 來預測分離目標訊號的強度時頻譜。

$$|\hat{S}| = |\hat{M}| \odot |X| \tag{2.6}$$

不同於直接預測強度時頻譜，強度遮罩描述了混音訊號及目標訊號間的關係。另外，在此方法中，不同的預測目標對映著不同的預測值域，例如預測目標為理想二元遮罩 (ideal binary mask, IBM) 時，$|\hat{M}|$ 中各元素的值為 0 或 1；預測目標為理想比例遮罩 (ideal ratio mask, IRM) 時，通常會假設目標訊號的強度低於混音訊號的強度，進而限制 $|\hat{M}|$ 中各元素值介於 [0, 1]，而這也是此類方法最常見的預測目標。

與章節 2.2.1 相同，我們需要使用混音訊號的相位 STFT 當作預測之相位，再通過等式 2.4、2.5 得到預測的分離訊號。

2.2.3 基於 cIRM 之模型預測

使用前面兩種方法的模型局限於強度的預測，而忽略了相位的重要性，單純使用混音訊號的相位無法代表聲源訊號的實際情況，尤其當殘響訊號比例越高時，越容易出現混音相位與聲源相位的重大偏差。Q. Kong [16] 等人統計了 MUSDB18 [12] 資料集中使用各種不同預測目標所能得到的最高 SDR 值，數據顯示失去相位的估計大幅限縮了模型的發展性。

相位的資訊固然重要，如何同時準確估算強度與相位卻是一大難題，這也成為了許多學者正在研究的主題。從許多文獻可以得知，單純利用相位時頻譜 (phase

doi:10.6342/NTU202303861

spectrogram) 來準確預測相位資訊非常困難 [17]，於是目前的方法通常會把强度資訊加入來協助相位的估算。相位的預測方法種類繁多，主要可分為直接預測、複數時頻譜 (complex spectrogram) 預測及基於 cIRM 預測。直接預測方法並未對相位做其它轉換而逕行預測，如在 PhaseNet [18] 中，作者將相位預測轉化成分類問題，先將相位定義於 $[-\pi, \pi]$ 區間後進行量化並搭配强度資訊進行估算。有些論文選擇使用複數時頻譜作為預測目標，此方法輸出實部與虛部，以複數平面的角度來看待强度及相位資訊。然而，直接預測複數時頻譜並不容易，因此另一種類似的做法是利用 cIRM 當作正確值做遮罩的估算，cIRM 可由下列下列公式得出:

$$\begin{aligned} M &= S/X \\ &= \frac{S_r + iS_i}{X_r + iX_i} \\ &= \frac{S_r X_r + S_i X_i + i(S_i X_r - S_r X_i)}{{X_r}^2 + {X_i}^2} \end{aligned} \tag{2.7}$$

利用此等式我們可以得出需要預測出的複數時頻譜遮罩，包含實部及虛部，此類方法在近期也有相當突出的效果。

2.2.4 强度與相位的分開預測

在 [16] 中，作者以另一個角度看待基於 cIRM 的預測問題，若以强度與相位的角度來看 cIRM，我們可以由以下等式得出聲源的資訊:

$$\begin{aligned} S &= MX \\ &= |M||X|e^{j(\angle M + \angle X)} \end{aligned} \tag{2.8}$$

藉由這一等式，我們可以利用對混音訊號的强度及相位變化來得到目標訊號的 STFT，於是問題從實部及虛部的預測轉化成了强度及相位的預測。雖然兩者的本質都是在預測 cIRM，但相較於直接預測複數時頻譜，這樣的方法似乎顯得更直觀，作者在論文中也提出這樣的預測目標增加了模型對於輸出設計的變化性，畢竟强度及相位兩者特性本就不同，針對其特性分別優化應當有助於模型的訓練。基於此構思，作者提出了預測强度及相位的設計方法。

在强度預測方面，公式 2.8 中的 $|M|$ 其實就是 2.2.2 中提及的强度遮罩概念。常見的方法總是限制遮罩的值域在 $[0, 1]$，這對於模型來說是把雙面刃，優點是讓模型預測大於 1 的遮罩容易產生嚴重偏差，所以這樣的設計有助於效果提昇，缺

doi:10.6342/NTU202303861

點則是會產生誤差，因為目標聲源强度可能會大於混音强度；根據 [16] 統計，在 MUSDB18 資料集當中，目標聲源訊號大約有 22% 的時候强度大於混音訊號，對於目標伴奏來說甚至高達 34.5%。為了同時解決以上問題，作者提出將强度遮罩 $\hat{M}_{mag}$ 與混音訊號强度做逐元素相乘後另外加上一項模型輸出 $\hat{Q}$ 得到目標聲源强度之預測:

$$|\hat{S}| = relu(\hat{M}_{mag} \odot |X| + \hat{Q}) \tag{2.9}$$

其中 $relu(*)$ 的使用是避免强度預測為負值。

在相位預測方面，我們可以利用複數遮罩的實部及虛部比值來得到聲源音訊從混合音訊改變的角度，此方法輸出 $\hat{P}_r$ 及 $\hat{P}_i$，分別代表複數遮罩的實部與虛部，藉此可以推導出正弦及餘弦值:

$$\begin{aligned} cos\angle\hat{M} &= \frac{\hat{P}_r}{\sqrt{\hat{P}_r{}^2 + \hat{P}_i{}^2}} \\ sin\angle\hat{M} &= \frac{\hat{P}_i}{\sqrt{\hat{P}_r{}^2 + \hat{P}_i{}^2}} \end{aligned} \tag{2.10}$$

利用得出的值，我們可以進一步利用和角公式得出聲源訊號的相位預測之正弦及餘弦值:

$$\begin{aligned} cos\angle\hat{S} = cos(\angle\hat{M} + \angle X) &= cos\angle\hat{M}cos\angle X - sin\angle\hat{M}sin\angle X \\ sin\angle\hat{S} = sin(\angle\hat{M} + \angle X) &= sin\angle\hat{M}cos\angle X + cos\angle\hat{M}sin\angle X \end{aligned} \tag{2.11}$$

至此，我們已經得到目標聲源的預測强度及正餘弦，可以直接計算其 STFT:

$$\hat{S} = |\hat{S}|e^{j\angle\hat{S}} \tag{2.12}$$

最後，再透過等式 2.5 即可得到預測的分離訊號。

2.3 注意力機制

卷積神經網路擷取局部特徵的特性，使之在影像辨識、訊號處理等任務皆獲得巨大的成功。單純以傳統卷積神經網路為主體的架構有許多優點，如參數量較多層感知器少、處理局部資訊的效果傑出等。然而，其最大的缺點就是受限於 kernel size 導致無法對全局資訊做進一步的分析。為了解決這棘手的問題，注意力

doi:10.6342/NTU202303861

(attention) 機制在近年來獲得廣泛的討論，尤其在自然語言處理 (natural language processing, NLP) 領域的 Transformer [19] 架構被提出之後，由於效果驚人，因此許多論文將其應用至不同領域 [20, 21]。

注意力機制的概念是對於當前關注的特徵，通過計算權重來加强與某個特定序列之間的關聯性。注意力機制主要分為四個步驟:

1. 對於當前關注特徵產生其 query 向量，對於目標序列產生 key、value 向量，如圖 2.4 所示。

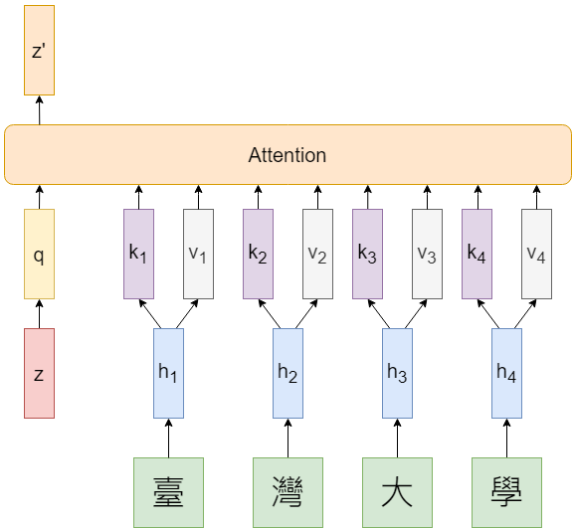


圖 2.4: 以計算當前特徵 z 對「臺灣大學」的注意力為例，生成當前特徵 query q、並從「臺灣大學」的隱藏向量 $h_i, i \in \{1,2,3,4\}$ 產生 key $k_i, i \in \{1,2,3,4\}$ 及 value $v_i, i \in \{1,2,3,4\}$。

2. 計算 query 與每個 key 的相似程度，我們可以使用 cosine similarity、點乘積運算等方法，如圖 2.5 所示。相似程度的值代表了當前特徵向量與序列中每個特徵向量之間的關係密切程度，數值越大代表關係越緊密。

3. 對於計算出的相似度取 softmax 得到當前特徵與目標序列的權重，如圖 2.6 所示。

4. 將權重值與各自的 value 相乘後相加，即得出當前特徵對目標序列的注意力，如圖 2.7 所示。

注意力機制主要分為兩種，若 query 與 key, value 為不同序列，我們稱其為交叉注意力 (cross-attention)，若為相同序列，則稱作自注意力 (self-attention)。在

doi:10.6342/NTU202303861

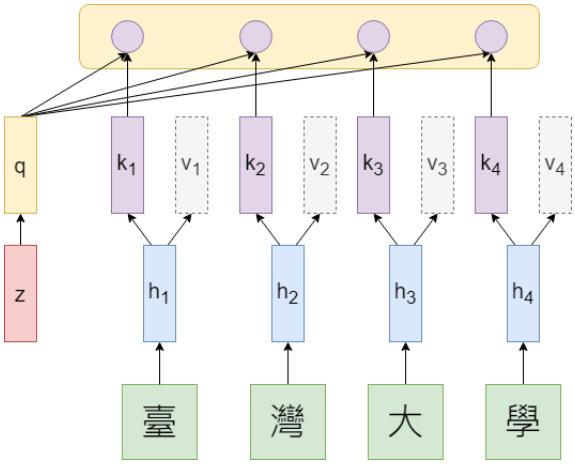


圖 2.5: 計算 q 與 $k_i, i \in \{1, 2, 3, 4\}$ 之間的相似程度

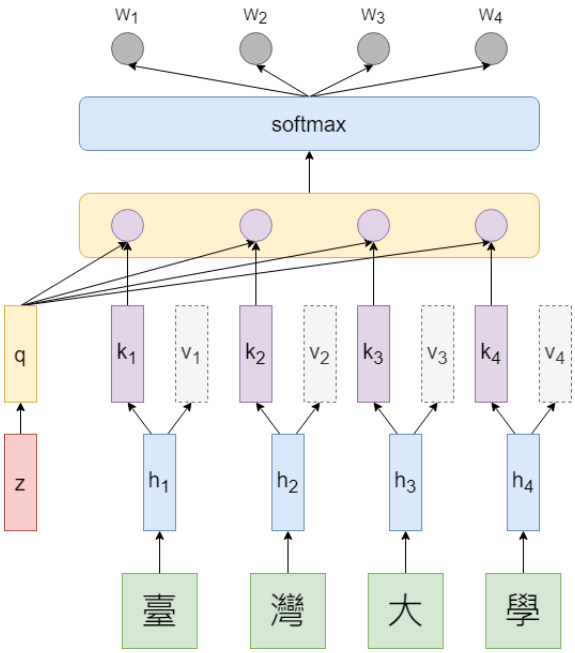


圖 2.6: 將相似程度值取 softmax 後得權重值 $w_i, i \in \{1, 2, 3, 4\}$

Transformer 架構中，自注意力使用於 encoder layer，作為特徵萃取的一環，而在 decoder layer 當中，交叉注意力可以讓模型利用 encoder output 的 query 找出較為重要的特徵當作輸出依據。

doi:10.6342/NTU202303861

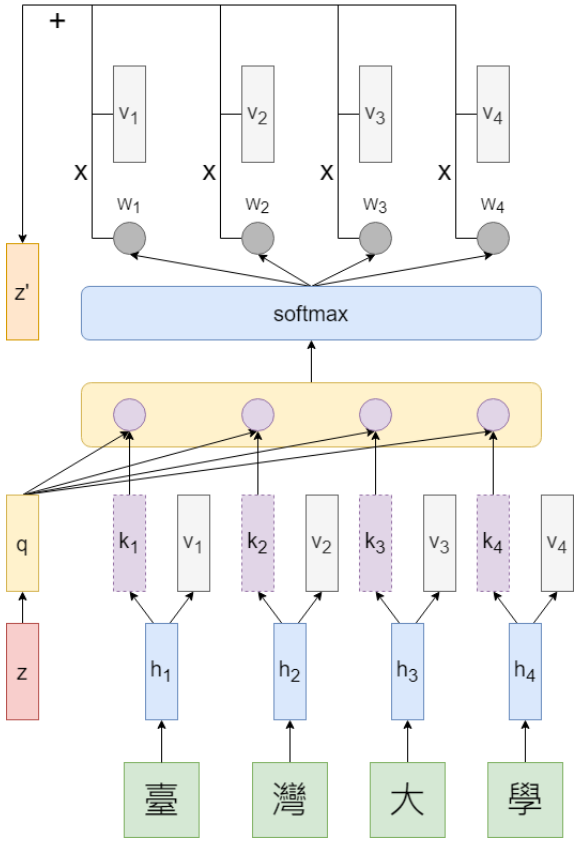


圖 2.7: 將權重值 w 與 v 逐個相乘並相加得當前特徵 z 對 h 的注意力 z'

2.3.1 自注意力機制套用至音樂聲源分離

在音樂聲源分離領域中，Y. Liu [22] 等人提出添加自注意力至 U-Net [23] 架構的神經網路進行分離，有助於提升模型表現，如圖 2.9 所示。對於時頻譜這類二維特徵輸入，該作者提出對時間的維度做自注意力，有助於讓當前時間軸的特徵考慮到未來和過去的資訊，進而提升表現。

除此之外，作者在注意力計算前的 query 及 key 添加了一層的線性層，以進行特徵萃取及維度壓縮，經過實驗後發現能加強模型的效果。詳細自注意力架構如圖 2.8 所示。

doi:10.6342/NTU202303861

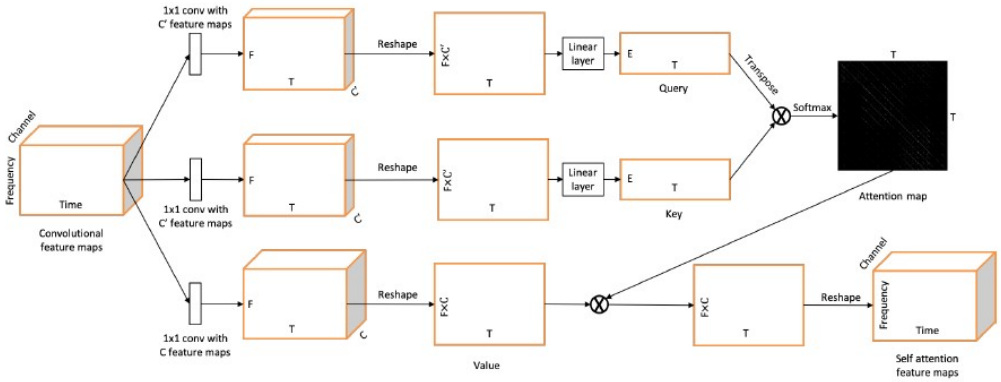


圖 2.8: Y. Liu 提出之自注意力架構 [22]

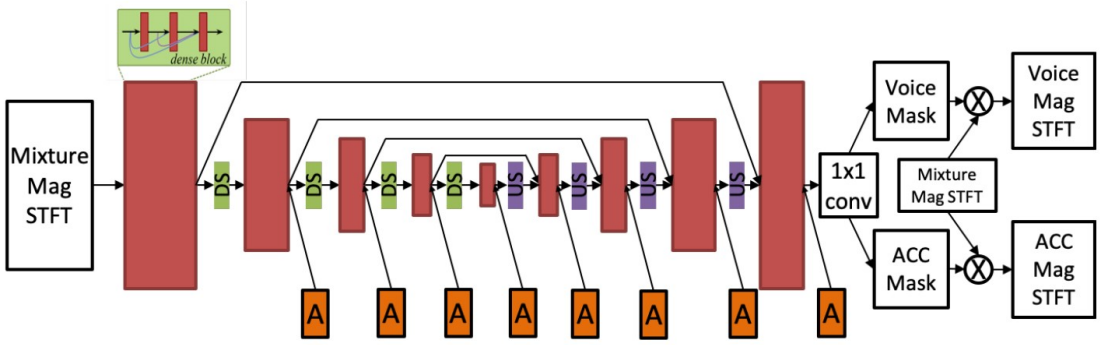


圖 2.9: Y. Liu 提出於 U-Net 套用自注意力之架構 [22]，圖中的 A 為添加之自注意力機制。

2.3.2 HT Demucs 架構簡介

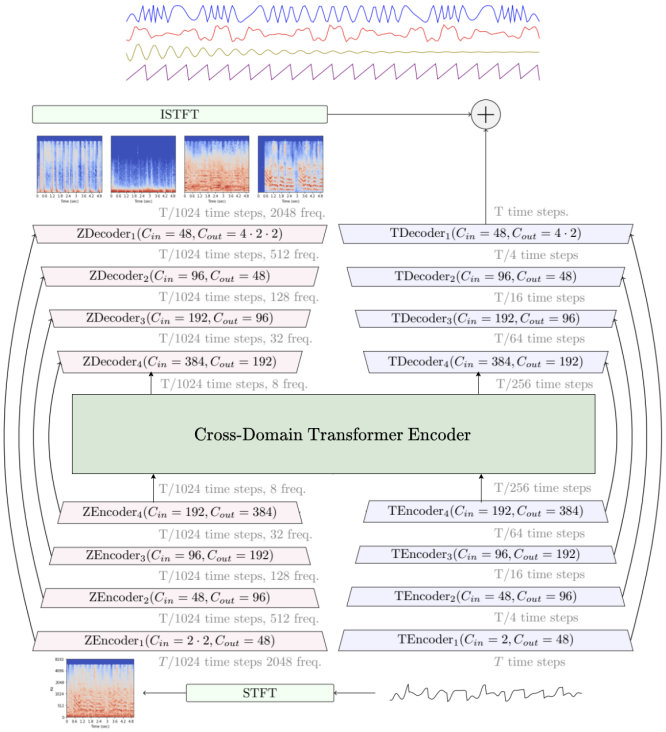
在所有音樂聲源分離的模型中，Meta AI 所提出的 Demucs 系列模型非常具有代表性，其最新版本的 Hybrid Transformer Demucs (HT Demucs) [2] 在 MUSDB18 測試集上為目前的 state-of-the-art (有添加額外的訓練資料)，架構如圖 2.10(a) 所示。初版的 Demucs [5] 為波形導向模型，從 WaveUNet [3] 進行延伸並在 bottleneck layer 使用 Bi-LSTM [6]，且於 decoder block 以轉置卷積替代線性插值。自 Demucs v3 (Hybrid Demucs) [11] 後，作者提出以兩個 U-Net [4] 分別輸入波形及時頻譜的資訊，為了讓兩種特徵進行訊息的傳遞，因此於 bottleneck layer 將兩個 U-Net 結合，並使用 Bi-LSTM [6] 和自注意力增加觀察視野。

doi:10.6342/NTU202303861

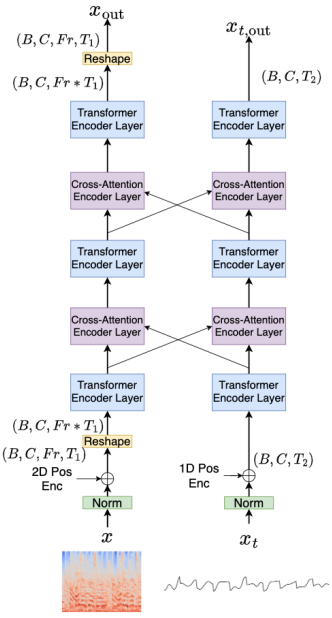
2.3.2.1 Cross-Domain Transformer Encoder

在最新的 HT Demucs [2] 中，bottleneck layer 的架構被 cross-domain Transformer Encoder 所取代。如圖 2.10(b) 所示，該架構由 Transformer encoder layer 和 cross-attention encoder layer 疊加而成。前者主要作用是捕捉當前序列的資訊，其主體為自注意力機制；後者則使用交叉注意力以進行跨序列的資訊傳遞。

相較於 Demucs v3 [11]，這樣的設計有許多優點。首先，不再需要仔細調整模型的參數以對齊波形及時頻譜的資訊。其次，實作起來更加簡單，並且可以輕鬆地更改架構的細節。因此，這種新的架構在音訊分離領域中具有潛在的應用價值。



(a) Hybrid Transformer Demucs [2]



(b) 深度為 5 層的 Cross-domain Transformer Encoder [2]

圖 2.10: Rouard, S. 等人提出的 Hybrid Transformer Demucs 架構細節。(a): Hybrid Transformer Demucs 於 U-Net 外面四層保留了 Hybrid Demucs 架構，並在中間使用 Cross-domain Transformer Encoder。(b):Cross-domain Transformer Encoder 由 Transformer encoder layer 和 cross-attention encoder layer 組成，同時運用了時域及頻域資訊。

doi:10.6342/NTU202303861

2.4 BSRNN 架構簡介

Band-split RNN (BSRNN) [9] 是另一個代表性的音樂聲源分離模型，其表現僅僅略遜於 HT Demucs [2]。BSRNN 為時頻譜導向的模型，輸入為複數時頻譜，訓練目標為目標聲源的複數遮罩。BSRNN [9] 的架構 (圖 2.11) 可以簡單分為三個區塊，分別為 band split module、band and sequence modeling module 及 mask estimation module。

2.4.1 Band Split Module

在混合聲源經 STFT 轉換為複數時頻譜後，接著進入 band split module。band split module 將複數時頻譜切分為 K 個不同的頻帶，並將不同頻帶送入不同的全連接層，各自進行運算後，每個頻帶會萃取出相同大小的特徵圖。這些特徵在 band split module 的最後一個階段合併成一個三維的特徵，並將其送入 band and sequence modeling module。

Band Split Module 的用意在於讓不同頻帶專注於學習各自的局部特徵，因此不同頻帶的切法將會影響最後的分離效果。

2.4.2 Band And Sequence Modeling Module

將不同頻帶的特徵合併之後，band and sequence modeling module 會將合併後的特徵進行兩次序列模型運算。模型沿著特徵中的時間軸及頻帶軸分別進行 layer normalization、Bi-LSTM、全連接層三個階段的計算，並使用跳躍連接的架構。特徵經過 band and sequence modeling module 後會維持原先的大小，接著送入 mask estimation module 進行最後的運算。

Band and sequence modeling module 主要在進行不同頻帶間特徵的相互萃取，這樣的做法能夠使不同頻帶觀察到其他頻帶的特徵，進而進行較為精確的模型預測。

2.4.3 Mask Estimation Module

Mask Estimation Module 首先將三維特徵切分回 K 個不同的頻帶特徵圖，不同的特徵圖各自運算不同的 layer normalization 及 MLP(multilayer perceptron) 層後，還原回原大小的頻帶。將不同的頻帶進行合併，即可得預測的目標聲源遮罩。

doi:10.6342/NTU202303861

Mask Estimation Module 主要是在進行最後遮罩預測的部分，受到 [24] 的啓發，作者使用 MLP 取代全連接層，以精準預測複數遮罩。

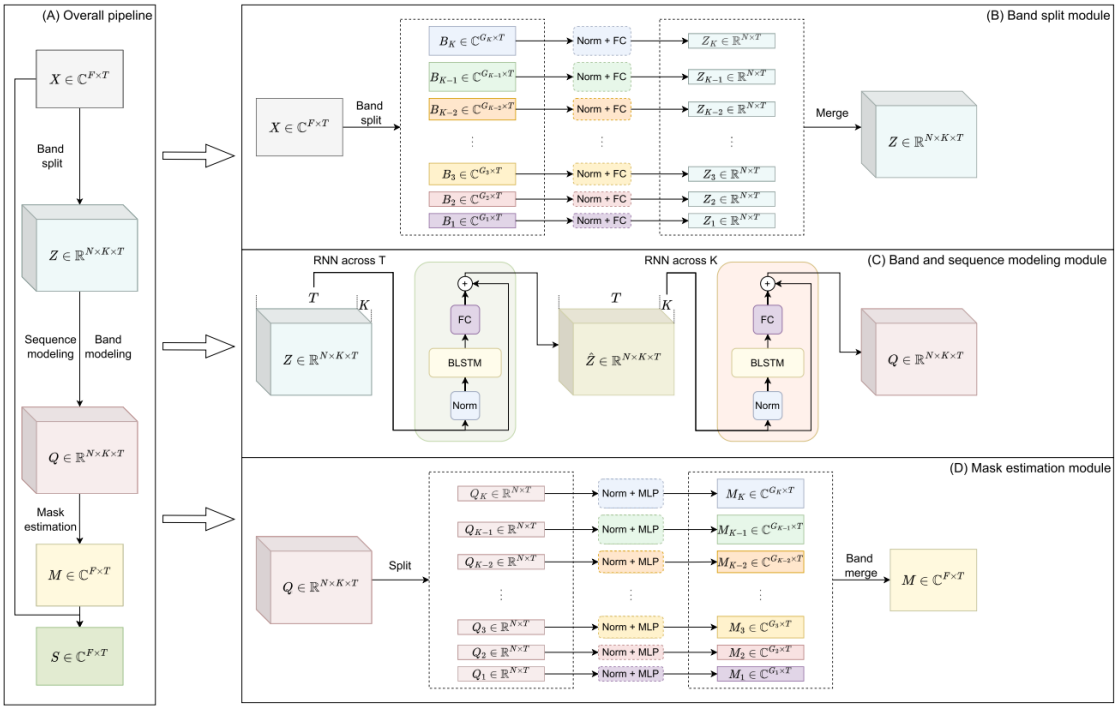


圖 2.11: BSRNN 模型架構圖 [9]。(a) BSRNN 完整流程示意圖，包含 band split module、band and sequence modeling module 及 mask estimation module。(b) Band split module 的架構設計，此模組會將複數時頻譜切為不同大小的頻帶。(c) Band and sequence modeling module 的架構設計，此模組對時間軸及頻帶軸做序列運算。(d) Mask Estimation Module，此模組為模型架構的最後一部分，會輸出預測的複數遮罩。

doi:10.6342/NTU202303861

第三章　研究方法

在本次音樂聲源分離的研究當中，我們使用 MMDenseNet 當作基準模型，並參考 Github 上的開源程式碼[1]進行初步的模型建置，我們使用 Pytorch 程式庫試圖在多個面向改進基準模型，並使用客觀指標評估方法的適用性。

3.1　問題定義

在本論文中，我們專注於音樂聲源分離中的伴奏分離，伴奏分離旨在將一首歌曲當中的非主要人聲部分進行萃取；在本次研究當中，我們希望提出的方法能在下列三項目標中取得平衡:

1. 基於基準模型提昇分離品質，目標為盡量逼近目前最佳模型的 SDR (signal-to-distortion ratio) 值。

2. 低延遲，最終目標為延遲時間低於 200 毫秒，但是此目標目前來說過於遙遠。因此在本論文中，會先以延遲時間接近 1 秒為階段性目標。此外，RTF (real time factor) 必須小於 1 以利於即時運算。

3. 低空間資源，目標為參數量小於 1M 以減少記憶體用量。

MMDenseNet [1] 是 2017 年所提出的模型，雖然分離品質無法與現今模型相比擬，卻擁有一個極為輕量化的架構，非常適合當作我們實驗基準模型，現今模型往往參數量過於龐大，在 edge device 下不是難以運行，就是運算速度過於緩慢。MMDenseNet 的輸入為一首歌曲的強度時頻譜 (magnitude spectrogram)，預測目標為聲源訊號的強度時頻譜，在本論文中，我們固定 MMDenseNet 的模型輸入為強度時頻譜，並將目標聲源訊號定為伴奏聲源。基於 MMDenseNet，我們提出了多種方法希望逼近上述三項目標:

[1]https://github.com/tky823/DNN-based_source_separation

doi:10.6342/NTU202303861

1. 基於理想比例遮罩 (ideal ratio mask, IRM) 與複數理想比例遮罩 (complex ideal ratio mask, cIRM) 調整模型訓練目標以提昇分離品質。
2. 調整模型架構以提昇分離品質，我們實驗了多種現有的自注意力機制並且提出可行的模型架構。
3. 提出新穎的漸進式測試及訓練方法在減少延遲的情況下維持模型的分離品質。

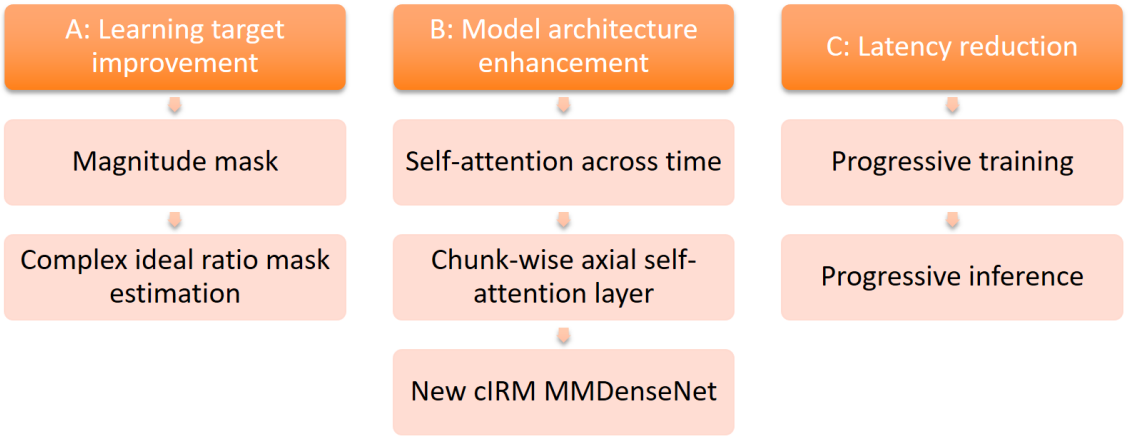


圖 3.1: 本論文提出之方法路線圖

3.2 MMDenseNet 模型分析

我們首先對 MMDenseNet 進行四項指標的觀察，這將大幅影響我們的優化方向，經由 3.1 可以發現，MMDenseNet 在「空間資源」這項目標已經達成，並且可以接受增加一定程度的 RTF 及參數量。在分離品質方面，我們可以發現其 SDR 距目標仍有一段距離，這將是我們優化的其中一個方向。我們得出的延遲時間為 4.38 秒，高於我們所訂定的低延遲標準。MMDesneNet 為全卷積神經網路 (fully convolutional network) 架構，所以模型輸入可以是任意長度的片段，而縮短長度能減少延遲時間。但是縮短長度時會受到卷積中 padding 的影響，進而降低分離品質。另外，進行後續實驗時使用的自注意力機制更是注重長時間的輸入片段，這也是我們必須克服的難題。

doi:10.6342/NTU202303861

表 3.1: MMDenseNet 評量指標初步觀察。SDR 為分離品質指標、latency 對應延遲大小、RTF 表示模型是否能進行實時分離、parameters 代表模型占用的空間資源多寡。

Architecture	SDR	Latency (sec)	RTF	Parameters (K)
MMDenseNet	11.162	4.38	0.3683	339

3.3 訓練目標調整

我們利用不同的模型訓練目標試圖改進模型的分離品質，不同的訓練目標代表模型在進行訓練時有不同的的真實值 (ground truth)；這類方法在幾乎不增加模型複雜度的情況下仍有機會提升模型的分離效果，非常適合本論文來進行使用，並於後續討論不同訓練目標的實驗結果。在原始的 MMDenseNet 當中，模型訓練目標為伴奏經過短時距傅立葉轉換 (short time fourier transform, STFT) 後的強度時頻譜。

3.3.1 強度遮罩

在語音增強 (speech enhancement) 及聲音分離等深度學習模型當中，我們常會看到論文使用維納濾波器 [25] (Wiener filter) 當作後處理 (post processing) 的手法，並通常展現不錯的去雜訊效果，然而，由於維納濾波需要模型預測出所有聲源來進行計算，運算過程較為複雜且繁瑣，並不適用於本論文，作為替代，我們讓模型預測強度遮罩，模擬類似於維納濾波的效果。

基於原 MMDensNet [1] 架構，我們利用理想比例遮罩取代強度時頻譜當作新的訓練目標，模型輸出從強度時頻譜轉變成整首歌曲的強度遮罩 (magnitude mask)，與原歌曲的強度時頻譜進行逐元素相乘當作伴奏的強度時頻譜預測，將此預測與原歌曲的相位 STFT 經過反短時距傅立葉轉換 (inverse short-time Fourier transform, iSTFT) 可得預測之伴奏訊號。

doi:10.6342/NTU202303861

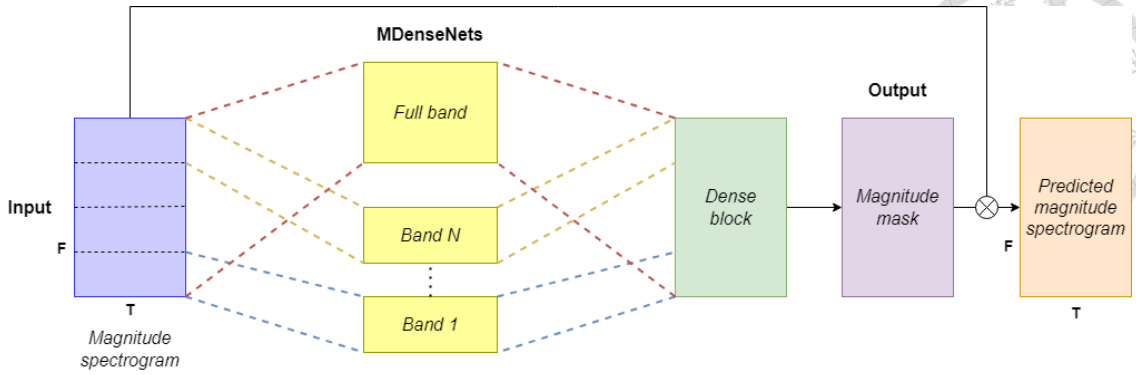


圖 3.2: 强度遮罩模型架構圖，歌曲强度時頻譜與輸出强度遮罩進行逐元素相乘後得模型預測的聲源時頻譜。

3.3.2 强度及相位的分開預測

前面提到的方法中，我們著重在時頻譜的强度預測，相位則是直接使用輸入聲源的資訊，這會使模型的預測產生誤差。為了讓模型能夠輸出相位資訊，本論文將 [16] 提出的方法嘗試套用至 MMDenseNet 中，並將這個版本的模型稱作 cIRM MMDenseNet。

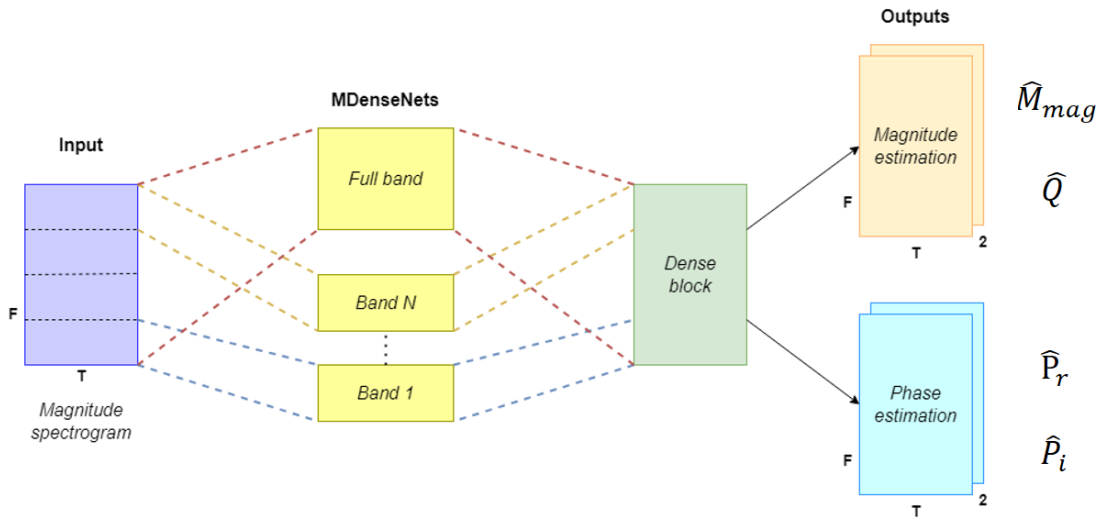


圖 3.3: cIRM MMDesneNet 模型架構圖，訓練目標為理想比例遮罩，經由分開預測强度及相位達成目標。

圖 3.3 為 cIRM MMDenseNet 的架構圖，其輸入與前述方法無異，不同的點在於 cIRM MMDenseNet 經由最後的 dense block 產生了四項輸出，分別為强度遮罩 $\hat{M}_{mag}$、直接强度預測 $\hat{Q}$、實部預測 $\hat{P}_r$ 及虛部預測 $\hat{P}_i$。前兩項用來估計强度時頻譜、後兩項經由計算角度後產生相位偏移量的預測，最後將强度及相位的預測

doi:10.6342/NTU202303861

經由反短時距傅立葉轉換可得預測之伴奏訊號。

3.4 模型架構調整

在原本的 MMDenseNet 模型中，可以發現其架構的主體為卷積神經網路 (convolution neural network, CNN)。CNN 能有效捕捉局部資訊，然而，純 CNN 的架構難以捕捉全局資訊，因此處理較長且序列化資料的效果較為遜色，例如文字、音樂、語音等等。近年來，由於自注意力機制的興起，許多研究將其套用至不同模型當中後，有效提升了表現。

本論文將探討不同自注意力機制添加於 cIRM MMDenseNet 對於分離效果的影響，除此之外，本論文調整了原架構並提出 New cIRM MMDenseNet，希望進一步提升伴奏的分離表現。

3.4.1 對時間軸的自注意力機制

基於 [22] 提出的方法，本論文將自注意力機制套用到 cIRM MMDenseNet 當中。Liu 等人將自注意力機制應用於音樂聲源分離領域，對於輸入的強度時頻譜，作者將其套用於時間軸上。在此架構的基礎下，本論文進行了以下三項調整，除了成功將自注意力機制應用至 MMDenseNet 上，也希望減少參數量及增加分離效果。

1. 對於不同頻帶的 MDenseNet，各自添加自注意力機制。由於音樂訊號由多種不同頻率的樂器所組成，因此不同頻帶間會產生巨大的差異，讓不同頻帶套用不同的自注意力機制除了易於實作，也能讓不同頻帶各自採用相對重要的時間段。

2. 結合 CNN 與自注意力機制的特性，實現多頭自注意力機制 (multi-head self attention)。本論文將 CNN 輸出的不同頻道 (channel) 當成不同的頭各自進行自注意力運算，此作法不會將不同頻道進行合併，因此可以減少線性層的參數量；此外，也能使不同頻道專注在各自的特徵上。

3. 為了讓訓練時更加穩定，本論文參考 [19] 的 Transformer 架構，套用 layer normalization 於自注意力機制當中。

doi:10.6342/NTU202303861

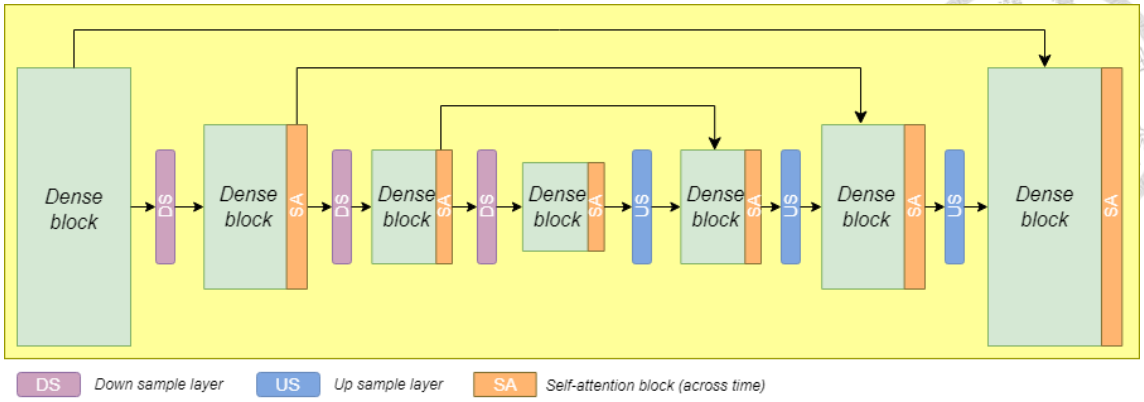


圖 3.4: 對時間軸的自注意力機制套用於單一 MDenseNet 模型架構圖

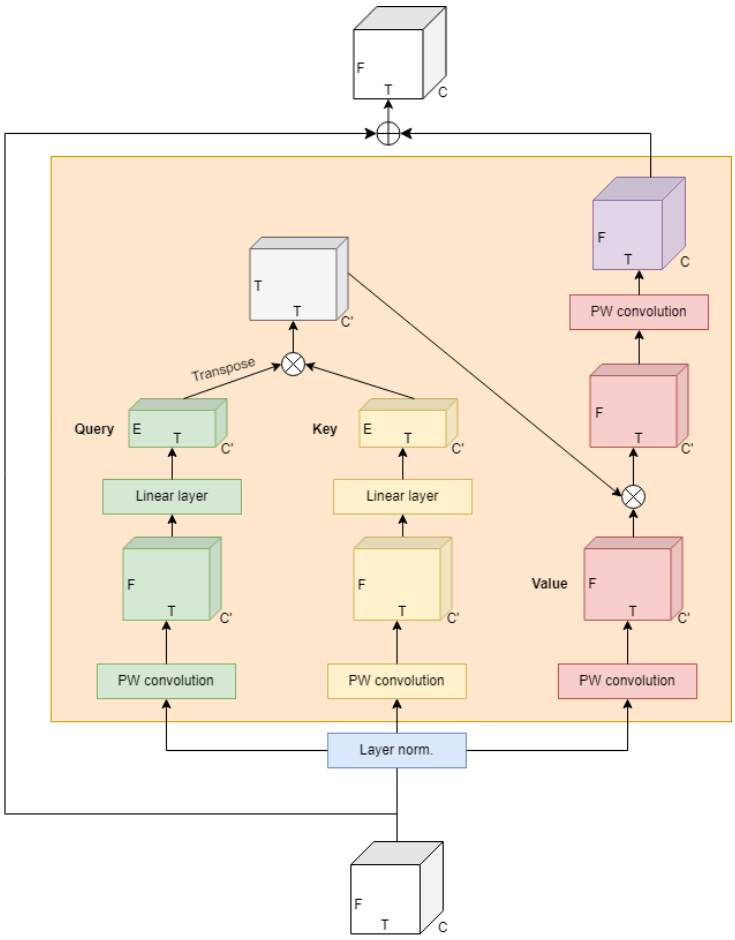


圖 3.5: 本論文調整過後的自注意力機制詳細架構圖，橘色部分為對時間軸的自注意力區塊，從左至右分別是自注意力的 query, key, value 區塊，每個自注意力機制都有作 skip connection 連接。

圖 3.4 為自注意力機制套用至 MDenseNet 的架構圖，這裡的設定參考 [22]，可以發現除了第一層外，在每個 dense block 後都添加了自注意力機制，對不同視野的特徵都觀察全局的資訊。

圖 3.5 為自注意力詳細架構圖，query、key、value 經過 pointwise convolution

doi:10.6342/NTU202303861

後讓原頻道數從 C 減少至 C'，對於這些頻道，我們做 C' 個頭的自注意力機制；經此調整過後，中間 linear layer 的參數量可以從 $F \times T \times E$ 減少為 $F \times E$，將參數減量少為 $1/T$ 倍。參數設定的部分，我們將 E 設為 20，並將 C' 統一設為 5。

3.4.2 軸向注意力機制

對於二維資訊，直接套用傳統的一維自注意力機制會導致計算複雜度過於龐大。過去曾有一些自注意力機制被提出應用於圖片當中，如 Huang [26] 提出的 criss-cross attention、Wang [27] 等人提出的軸向注意力 (axial attention) 等。本論文以軸向注意力為研究主軸，原先軸向注意力被提出應用於全景分割 (panoptic segmentation) 任務，主要作法是對二維資訊的兩個軸各自進行自注意力機制，在模型架構當中，結合兩軸的自注意力後，可以讓圖片中任一點觀察到全局的資訊。此外，比起將圖片攤平成一維再套用自注意力機制，此作法能減少時間複雜度。

在音樂相關任務的領域當中，對自注意力的研究較為稀少，本論文以軸向注意力為借鏡，提出一種可以應用於較短時間的新機制。其主要作法包含了上一節提出的對時間軸的自注意力機制及接下來將提到的對頻率軸的區塊式自注意力機制。

3.4.2.1 對頻率軸的區塊式自注意力機制

在本論文中，為了善加利用自注意力機制能夠觀察寬廣視野的特性，提出對頻率軸使用區塊式自注意力機制。其架構大多與對時間軸的自注意力相同，但是為了在測試階段能夠以相較訓練階段較短的秒數進行伴奏分離，本論文提出區塊式自注意力 (chunk-wise self-attention) 作法 (如圖 3.6 所示)，將輸入根據時間軸切分為大小為 $1 \times t$ 的區塊，並對同時間軸的頻帶使用自注意力機制，因此對於一個長度為 T 的時頻譜來說，總共會做 T/t 個不同的多頭自注意力機制。

圖 3.7 為對頻率軸的區塊式自注意力的詳細架構圖，圖中的 N 代表的值為 T/t。對於大小為 $F \times T \times C$ 的輸入來說，會先將其經過 pointwise convolution 縮小成 $F \times T \times C'$；接下來會根據區塊長度 t 做基於頻率軸的區塊式自注意力機制。在本實驗的設定當中，t 將設置為 16。此模型架構將應用於軸向注意力機制中的頻率軸自注意力區塊。

doi:10.6342/NTU202303861

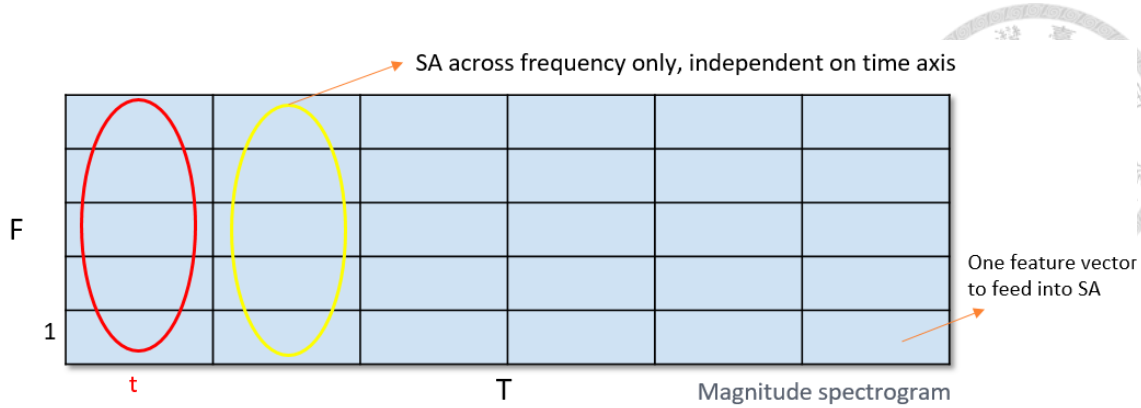


圖 3.6: 區塊式注意力機制將長度為 T 時頻譜切為每 t 個一組，再各自沿頻率軸做自注意力。

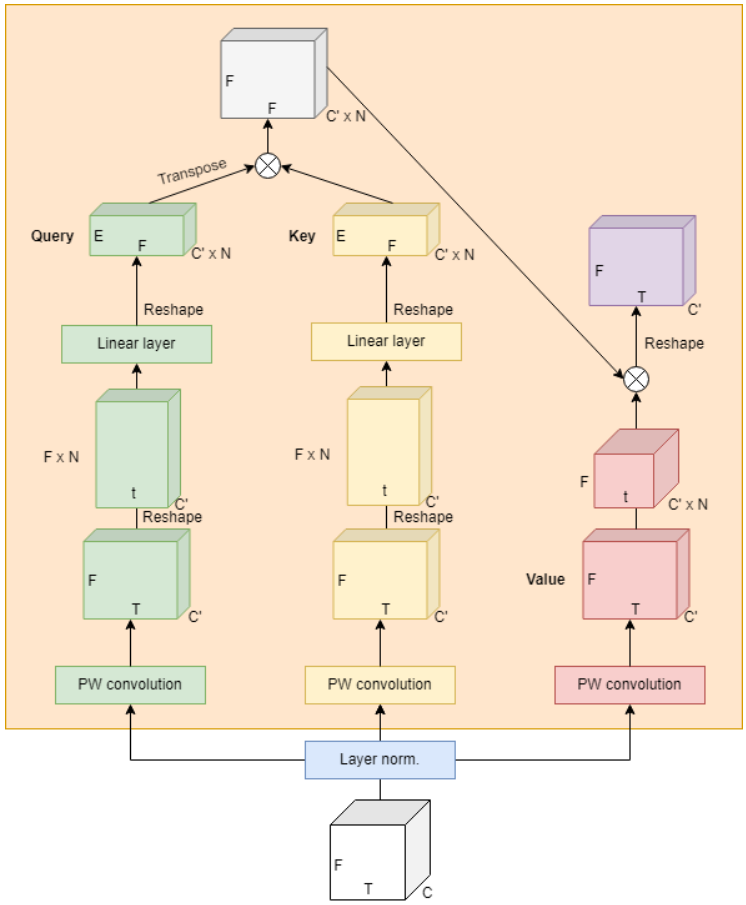


圖 3.7: 本論文提出的區塊式注意力機制詳細架構圖，橘色部分為對頻率軸的自注意力區塊，從左至右分別是自注意力的 Query, Key, Value 區塊，圖片中的 N 對應的是輸入長度 T 共被切為多少個長度為 t 的區塊。

3.4.2.2 應用軸向自注意力機制於 cIRM MMDenseNet

將上述提到的對時間軸的自注意力機制及對頻率軸的區塊式自注意力機制做結合，形成本論文所提出的軸向注意力機制，並將其應用於伴奏分離任務 (如圖

doi:10.6342/NTU202303861

3.8 所示)。在實驗中，我們從 cIRM MMDenseNet 做延伸，將軸向自注意力套用於 full band MDenseNet 當中，其他頻帶的 MDenseNet 則是單純使用對時間軸的自注意力機制。之所以做這樣的設定，原因是希望 full band MDenseNet 能夠轉注在其餘 subband MDenseNet 沒辦法考量的全局頻率特徵，且這樣的設定對參數量及運算量的影響較小。

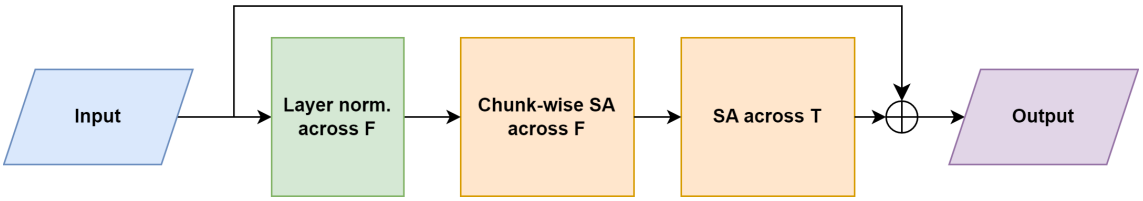


圖 3.8: 軸向注意力應用於 full band MDenseNet 架構圖，經過 layer normalization 後，會先進行頻率軸的區塊式自注意力，接著套用時間軸的自注意力，最後再用 skip connection 與輸入做加總得到輸出。

3.4.3 New cIRM MMDenseNet

觀察 BSRNN [9] (band-split RNN) 與 MMDenseNet [1] 的模型架構，可以發現兩者的相似之處。BSRNN 中的 band split module，與 MMDenseNet 中不同頻帶的 MDenseNet，目標都是希望萃取不同頻帶獨有的特徵。此外，兩者也都擁有計算目標遮罩的區塊。但是在 MMDenseNet 當中，我們找不到與 BSRNN 中 sequence and band modeling module 相對應的架構，sequence and band modeling module 中的 Bi-LSTM 架構能夠捕捉重要的跨時間資訊與跨頻帶資訊，進而讓每個特徵的觀察視野變得寬廣。受到 BSRNN 的啓發，本論文調整了 cIRM MMDenseNet，並提出了一個能讓不同頻帶的 MDenseNet 相互捕捉資訊的架構，是為 New cIRM MMDenseNet。

New cIRM MMDenseNet 在 cIRM MMDenseNet 的基礎下，新增了不同 MDenseNet 在 bottleck layer 的相互連接。其主要架構如圖 3.9 所示，主要分為三個階段，分別為 band merge，cross-band axial attention 及 band split。

3.4.3.1 Band Merge

在 band merge 階段，我們將不同頻帶中 MDenseNet 的 bottleneck 特徵相互連接，以便後續進行所有頻帶特徵間的運算。MDenseNet 根據不同的初始設定，其 bottleneck 特徵的形狀會因此改變。為了進行頻帶的合併，必須使不同特徵間

doi:10.6342/NTU202303861

的 channel 數相等。假設我們將強度時頻譜切分為 K 個不重疊的頻帶，則會在 bottleneck layer 產生 K 個不同大小的三維特徵 P,

$$P_i = F_i \times T \times C_i, i \in \{1, 2, ..., K\} \tag{3.1}$$

其中 F_i 為頻率軸長度、T 為時間軸長度、C_i 為 channel 數量。接著我們取最大的 channel 數為 C,

$$C = max(C_1, C_2, ..., C_K) \tag{3.2}$$

並使用 batch normalization 及 pointwise convolution 的運算將所有 P 的 channel 數增加為 C，對於原先 channel 數就已經為 C 的特徵則是不做處理。

$$P_i' = \begin{cases} PWConv(Batchnorm(P_i)), & \text{if } C_i < C \\ P_i, & \text{otherwise} \end{cases}, i \in \{1, 2, ..., K\} \tag{3.3}$$

最後就能把所有的 P_i' 沿著頻率軸合併為一個完整的三維特徵。

3.4.3.2 Cross-band Axial Attention

將特徵合併之後，我們使用 3.4.2 中提出的軸向注意力機制進行跨頻帶間的資訊共享。輸入特徵首先經過 layer normalization，接著對頻率軸進行區塊式自注意力機制，這一部分能讓每個特徵點萃取對其較為重要的頻率；隨後對時間軸進行自注意力機制，使特徵專注在較為重要的時間段。最後使用跳躍連接的架構，將 cross-band axial attention 前後的特徵加總後得到輸出。

3.4.3.3 Band Split

在 band split 階段，我們將處理過的特徵拆分回原輸入大小的不同頻帶特徵 $P_i', i \in \{1, 2, ..., K\}$；這些特徵未來會做為不同 MDenseNet 中 decoder 的輸入 $Q_i, i \in \{1, 2, ..., K\}$。在這個階段，我們需要將 channel 數 C 還原回 C_i，做法與 band merge 階段類似，都是使用 batch normalization 及 pointwise convolution 的運算將 channel 數還原為 C_i，對於 $C_i = C$ 的特徵則是不做處理，可以直接當作

doi:10.6342/NTU202303861

decoder 的輸入。

$$Q_i = \begin{cases} PWConv(Batchnorm(P_i')), & \text{if } C_i > C \\ P_i', & \text{otherwise} \end{cases}, i \in \{1, 2, ..., K\} \quad (3.4)$$

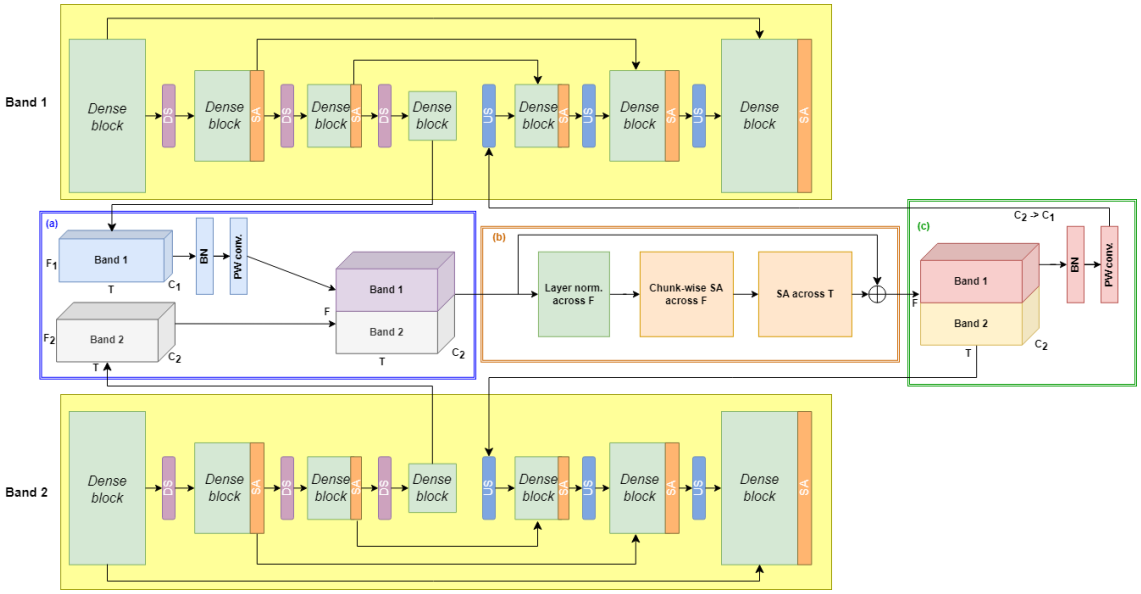


圖 3.9: New cIRM MMDenseNet 架構改進示意圖，這裡假設 $K = 2$，並且 $C_2 > C_1$。(a) Band merge 階段進行頻帶間的合併，由於 $C_1 < C_2$，所以必須經過運算將 C_1 增加至 C_2，以便進行合併。(b)Cross-band axial attention 為實際進行跨頻帶特徵運算的階段，使用軸向注意力進行特徵運算。(c)Band split 階段將處理過的特徵拆分回各自的 decoder，由於 $C_1 < C_2$，所以必須經過運算將第一個頻帶特徵的 C_2 還原回 C_1。

3.5 減少分離延遲

為了能夠進行即時伴奏分離，模型在進行低秒數的輸入時也必須要能有好的分離表現，然而大部分的模型無法適應縮短測試秒數的作法，尤其是參數量較少的模型，或是特徵觀察視野較為寬廣的架構如注意力機制。較為簡單的改進作法是直接將過去的音樂片段與當前要分離片段進行合併當作模型的輸入，然而這種做法會讓模型的分離時間大幅增加，並不符合我們需要實時分離伴奏的標準。

對於我們使用的 cIRM MMDenseNet 加上對時間軸的自注意力架構，本論文受到模型輸入越長分離表現越好的啓發，提出了兩種於測試時能夠使用的漸進式測試 (progressive inference) 方法，並將其中一種轉化為漸進式訓練 (progressive

doi:10.6342/NTU202303861

training) 方法。此外，本論文提出的方法可以彈性化的應用於多數 U-Net 架構當中。

我們首先定義以下三個術語以便後續進行解釋:

- Training segment: 訓練時，從資料集取出的歌曲片段強度時頻譜。
- Training chunk: 訓練時，輸入模型中的強度時頻譜。
- Test chunk: 測試時，輸入模型中的強度時頻譜。
- Look back chunk: 進行漸進式訓練及測試時，使用的過去特徵資訊。

過去的方法中，training chunk 及 test chunk 幾乎都在 3 秒以上。為了減少分離延遲，本節方法會縮短 training chunk 及 test chunk 至接近 1 秒。

3.5.1 U-Net 特化漸進式測試

本章節將介紹兩種漸進式測試方法，讓模型在測試時除了專注在當前的音樂片段，也能夠利用過去的資訊來進行預測。根據不同的特徵連接方式，本論文將提出的方法分為 look-back through bottleneck 及 look-back through encoder blocks。比起直接把過去的音樂片段當作輸入，這兩種方法的分離速度較快，能有較低的 RTF (real time factor)。

3.5.1.1 方法一: Look-back Through Bottleneck

對於 U-Net [4] 架構的模型，若要以最小的特徵維度代表模型輸入的一個音樂片段，我們可以使用 bottleneck layer 的特徵做為代表。因此，我們提出在模型進入當前的 bottleneck layer 前，加入過去音樂片段的 bottleneck 資訊進行運算，方法如圖 3.10 所示。

第一步，我們將整首歌曲的強度時頻譜根據時間軸切成 N 份 test chunk，對於每個強度時頻譜 $|X_i|$，不同的頻帶會由不同的 MDenseNet 進行運算。假設頻帶數量為 B，我們定義 $F_{i,j}$ 為第 i 份強度時頻譜的第 j 個頻帶中 MDenseNet 進入 bottleneck layer 前的特徵。第二步，我們假設模型往回看的 look back chunk 長度為 n，則 bottleneck layer 的輸入 $F'_{i,j}$ 會是前 n 個特徵與當前特徵在時間軸的合併:

$$F'_{i,j} = [F_{i,j-n}, F_{i,j-n+1}, ..., F_{i,j}] \tag{3.5}$$

doi:10.6342/NTU202303861

其中 [...] 為對時間軸合併的操作。

第三步，我們將 $F'_{i,j}$ 送入 bottleneck layer 經過運算得到 $O_{i,j}$。最後，由於 MDenseNet 的 encoder 與 decoder 間有跳躍連接，所以 $O_{i,j}$ 必須切回 $F_{i,j}$ 的大小，這就是整個 look-back through bottleneck 的流程。

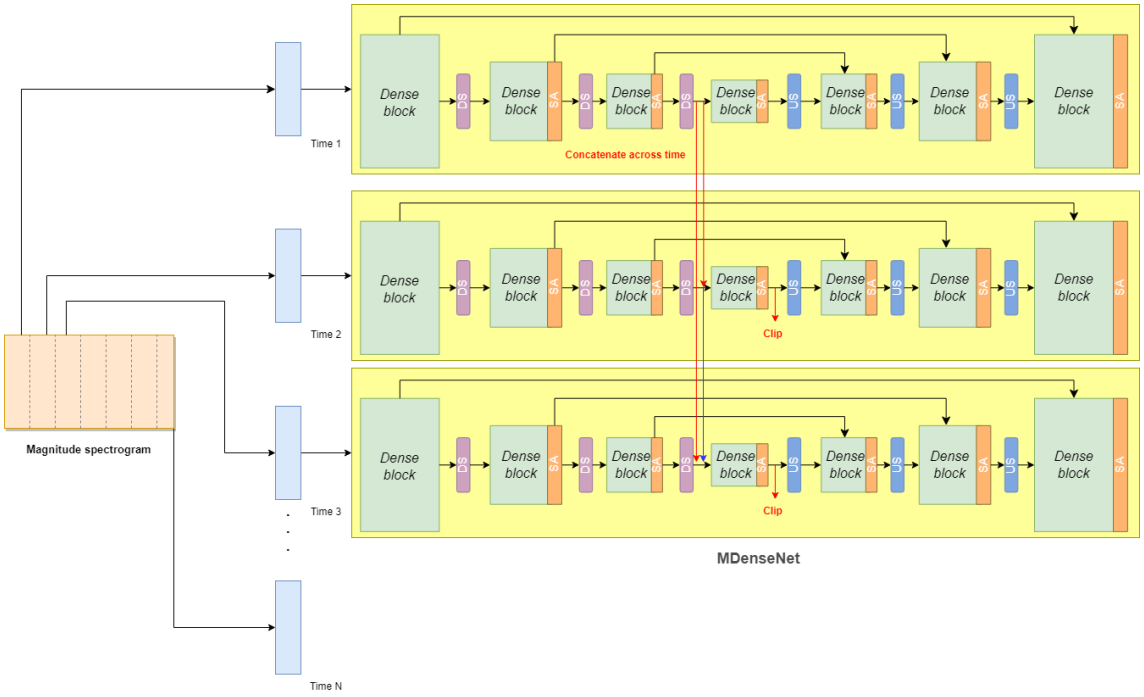


圖 3.10: 將 look-back through bottleneck 套用於 cIRM MMDenseNet 加上對時間軸的自注意力架構。圖中我們將 n 定為 2，由於 Time 1 的強度時頻譜沒有過去的特徵可以使用，所以維持初始 MDenseNet 的運算。Time 2 時能夠使用 Time 1 的特徵資訊進行 bottleneck layer 的 dense block 及自注意力運算，Time 3 則能夠使用 Time 1 及 Time 2 的資訊。

3.5.1.2 方法二: Look-back Through Encoder Blocks

在方法一當中，過去的特徵在 bottleneck layer 後就必須被截斷，於是我們提出了 look-back through encoder blocks 試圖讓過去的資訊在 decoder 中能繼續被善加利用。Look-back through encoder blocks 需要儲存過去的 encoder 輸出，藉此來減少運算時間，方法如圖 3.11 所示。

在方法二中，我們額外記住前 n 個時頻譜在所有 down sample layer 前的輸出 $E_{i,j,k}, k \in \{1, 2, ..., r\}$，其中 r 為 down sample layer 個數，k 越大代表越接近 bottleneck layer。接著，將過去的輸出進行合併，就能得到當前 decoder 需要用到

doi:10.6342/NTU202303861

的所有跳躍連接的輸入:

$$E'_{i,j,k} = [E_{i,j-n,k}, E_{i,j-n+1,k}, ..., E_{i,j,k}] \tag{3.6}$$

方法二延續方法一的前三個步驟。接著，我們將 $O_{i,j}$ 直接送入 decoder 的部分並在跳躍連接時使用過去的 encoder 資訊 $E'_{i,j,k}$。最後，在 MDenseNet 輸出時將時間軸切回與輸入時一樣的長度，即完成 look-back through encoder blocks。

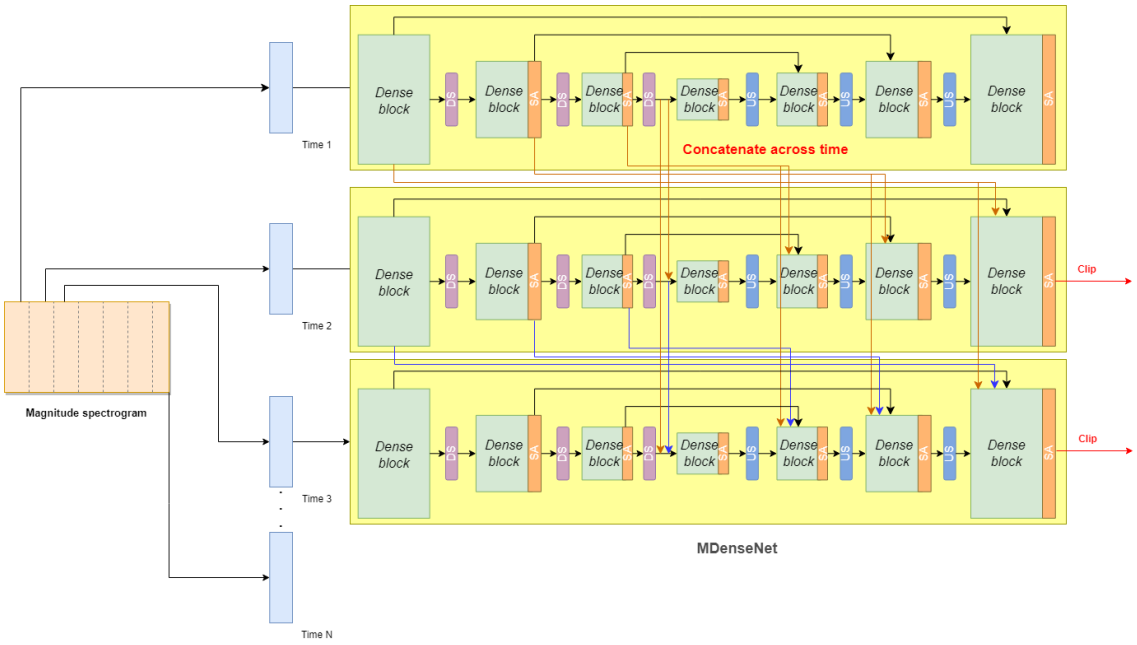


圖 3.11: 將 look-back through encoder blocks 套用於 cIRM MMDenseNet 加上對時間軸的自注意力架構。圖中我們將 n 定為 2，在 Time 2 中，MDenseNet 的 decoder 會得到 Time 1 中 encoder 的資訊，藉此來進行跳躍連接。Time 3 時也能考慮 Time 2 及 Time 1 的 encoder 資訊。此外，由於 Time 1 的强度時頻譜沒有過去的特徵可以使用，所以維持初始 MDenseNet 的運算。

3.5.2 U-Net 特化漸進式訓練

根據 3.5.1.1 提出的測試方法，本論文提出了漸進式訓練的做法，希望能讓模型在進行漸進式測試時有較好的分離效果。我們首先先定義以下三個術語:

漸進式訓練的流程如圖 3.12 所示，步驟與 3.5.1.1 大致相同，不過在進行訓練時，我們將資料切成相同大小的 training segment，並將其分為 C 個相同大小的 training chunk，因此每個 training segment 會使模型訓練 C 次。另外，我們選擇保留 look back chunk 長度不足的 training chunk 進行訓練。其原因有二，首先，我們希望模型在缺少 look back chunk 時也有能力進行分離；再者，這樣的設定可

doi:10.6342/NTU202303861

以有效增加訓練效率及方便性。

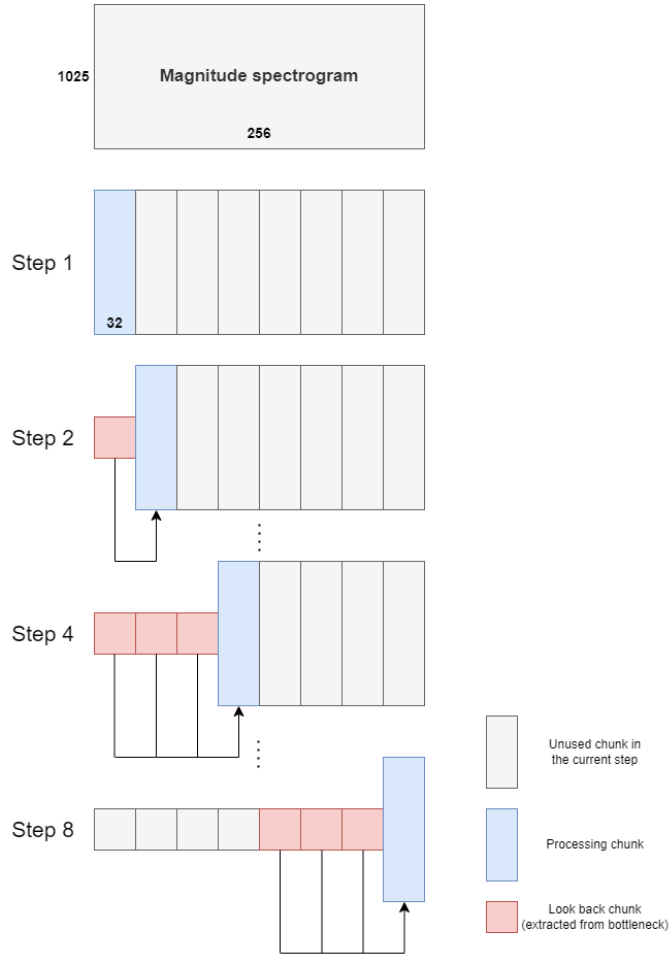


圖 3.12: 漸進式訓練流程，圖中我們使用大小為 1025 frequency bins × 256 time frames 的强度時頻譜當作 training segment，並將 training chunk 時間維度定為 32 time frames、look back chunk 最長時間維度定為 96 time frames。每一步當中，藍色區域代表模型當前預測的區塊，紅色區域代表模型當前使用的過去特徵，灰色則代表未使用的區塊。訓練 Step 1 到 Step 3 時，我們將 look back chunk 分別設定為設定為 0、32、64，其餘皆為 96。

doi:10.6342/NTU202303861

doi:10.6342/NTU202303861

第四章　資料集與實驗設定

4.1　資料集簡介

本節將介紹實驗中所使用到的資料集，我們使用的資料集為 MUSDB18 [12]，並且會依序介紹其子資料集，分別為 DSD100 [13] 及 MedleyDB [28]。

4.1.1　MUSDB18

MUSDB18 [12] 資料集起源於 2018 年所舉辦的 SiSEC (Signal Separation Evaluation Campaign) 音訊分離比賽 [29]，是一個用於音樂聲源分離的公開資料集。該資料集由 150 首雙聲道歌曲組成，分別來自不同曲風的歌曲及音樂家，其取樣頻率皆為 44.1kHz。圖 4.1 為曲風數量分布，可以發現 MUSDB18 資料集當中由流行搖滾樂為大宗，占比接近一半，其餘曲風包含重金屬歌曲、饒舌歌曲、爵士樂、鄉村音樂、電子音樂等。資料集中每首歌包含了四個聲源音軌，分別為人聲 (vocals)、鼓聲 (drums)、貝斯 (bass)、其他 (other)，四個音軌相加後即為混音軌 (mixture)，而本論文的分離目標為伴奏軌 (accompaniment)，由鼓聲、貝斯及其他相加組成。

在 MUSDB18 的 150 首歌曲當中，官方取其中 100 首做為訓練資料集，總歌曲長度約為 6.5 小時，使用者通常會取其中的 14 首當作驗證資料集；剩餘 50 首則為測試資料集，總長度約 3.5 小時。其資料來源包含 DSD100 中 100 首、MedleyDB 中 46 首、Native Instruments 2 首、加拿大搖滾樂團 The Easton Ellises 2 首。

doi:10.6342/NTU202303861

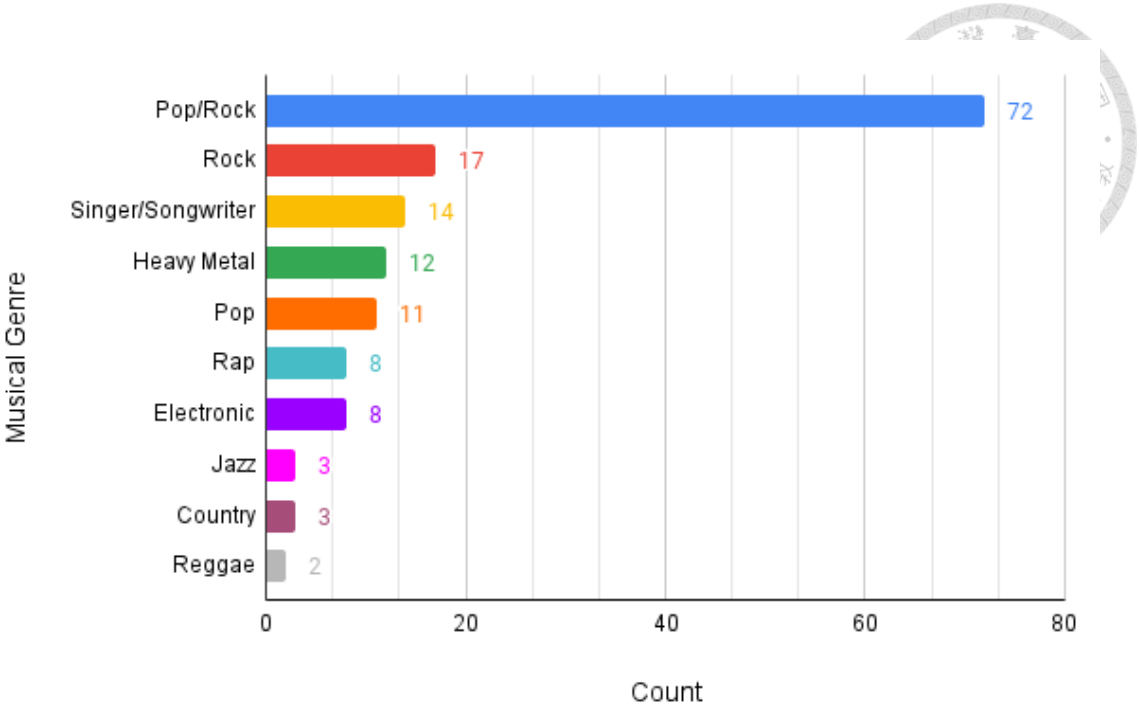


圖 4.1: MUSDB18 資料集歌曲曲風數量統計 [12]

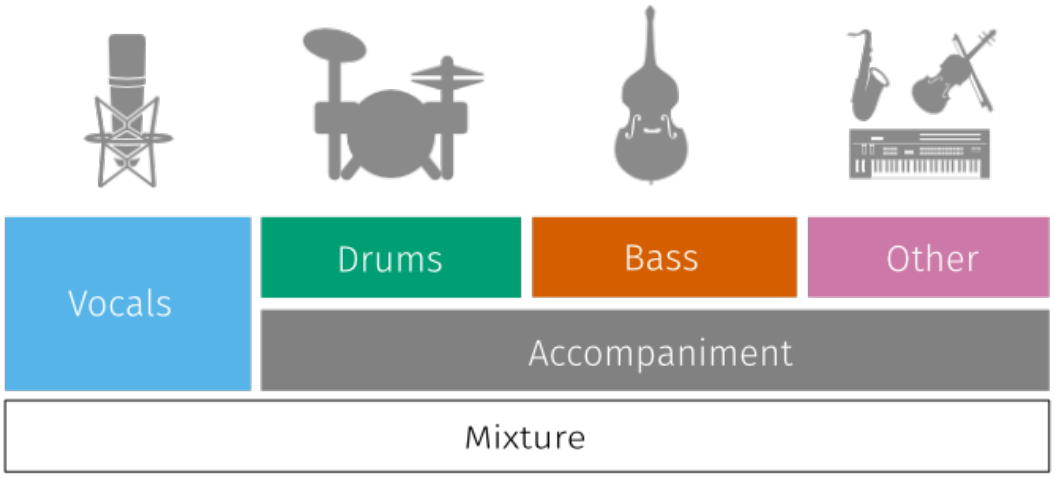


圖 4.2: MUSDB18 資料集聲源分部組成 [12]

4.1.2 DSD100

DSD100 [13] 為 MUSDB18 中的子資料集，起源於 2016 年所舉辦的 SiSEC 音訊分離比賽 [13]，前身為 MSD100 資料集，經專業的數位音樂工作站 (Digital Audio Workstation, DAW) 進行混音後而得，由曲風各異的雙聲道歌曲所組成，是一個公開資料集；在資料集的 100 首歌當中，官方將其中 50 首當作訓練資料、

doi:10.6342/NTU202303861

剩下的 50 首為測試資料；資料集中每首歌包含了四個聲源音軌，分別為人聲、鼓聲、貝斯、其他。此外，DSD100 是從「The 'Mixing Secrets' Free Multitrack Download Library」蒐集而來，為書籍「Mixing Secrets For The Small Studio」的衍生專案。

4.1.3 MedleyDB

MedleyDB [28] 由紐約大學音樂與聲音研究所 (Music and Audio Research Lab, MARL) 所主導發布，為 196 首多軌錄音所組成的音樂資料集，包含版本一的 122 首及版本二新增的 74 首[1]。資料集中包含了包括流行音樂、古典音樂等多種曲風的多軌音樂。這些音樂都是由真實的樂手演奏錄製而成，並由音樂專業人員進行混製；資料集中詳細標註每首歌的分軌、歌手資訊，以及其他音樂相關的資訊，如歌詞、節奏、旋律、曲風等。MedleyDB 資料集總共有超過 70 種不同的樂器分軌，相較於 MUSDB18 及 DSD100，MedleyDB 的標注較為詳細，但這也造成了一些標注錯誤或是樂器分軌無聲的情況發生。

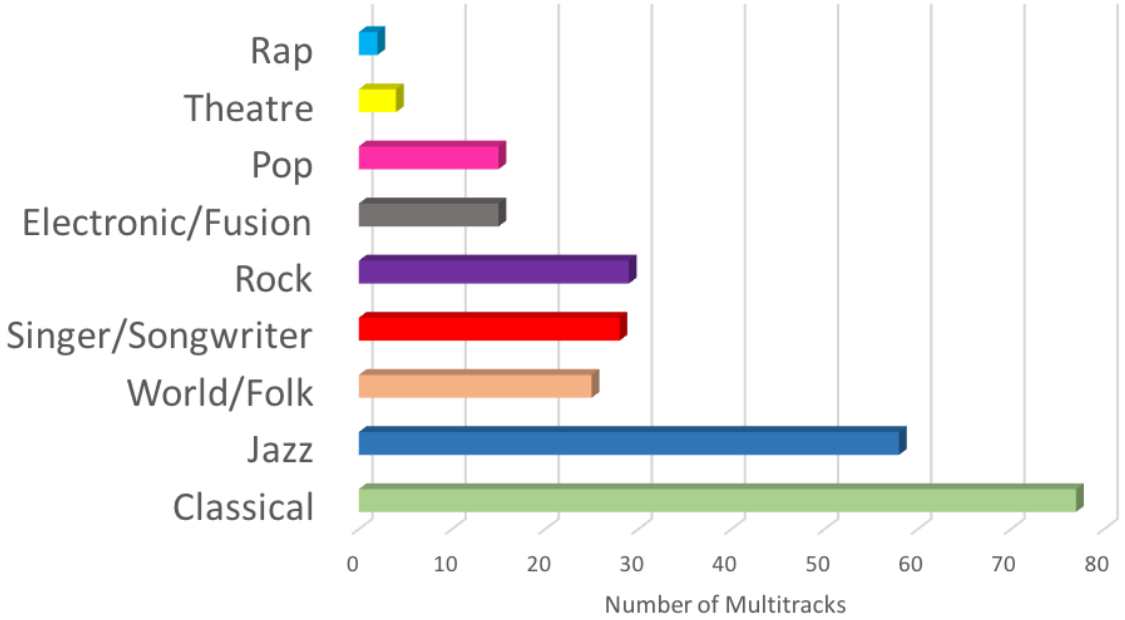


圖 4.3: MedleyDB 資料集歌曲曲風數量統計 [28]

[1]原論文內文中為 132 首，於官方網站說明當中則為 74 首，實際可從網路取得到的資料集曲目數為 74 首。

doi:10.6342/NTU202303861

4.2 評量指標

本論文希望在伴奏分離任務中，取得分離效果、延遲時間、空間資源三者的平衡。為了計算分離效果，我們採用常見的音樂聲源分離指標 SDR (Source-to-Distortion Ratio)；我們以 RTF (real-time factor) 及 test chunk 計算延遲時間的長短；空間資源則是以參數量作為評量指標。

4.2.1 SDR

Vincent [30] 等人提出了多種指標用來評估盲蔽訊號源分離 (blind source separation, BSS) 的好壞。論文中設計了兩個步驟來計算預測聲源 $\hat{s_j}$ 與目標聲源 s_{target} 間的關係。第一步，Vincent 定義預測聲源由目標聲源與三種誤差加總而得:

$$\hat{s_j} = s_{target} + e_{interf} + e_{noise} + e_{artif} \tag{4.1}$$

等式當中的 e_{interf}，e_{noise}，e_{artif} 分別代表干涉誤差、噪音誤差、人為干擾誤差。第二步，我們可以根據上述定義計算 SDR 值:

$$\text{SDR} := 10\log_{10}(\frac{||s_{target}||^2}{||e_{interf} + e_{noise} + e_{artif}||^2}) \tag{4.2}$$

在音樂聲源分離任務中，通常大多數研究都是使用 SDR 來評估預測聲源的品質。在我們的實驗當中，s_{target} 為伴奏訊號，s_j 為模型預測的伴奏訊號。根據 Vincent 等人的實驗，當 SDR 越高時，代表模型的總失真程度越低，通常也就意味著分離效果越好。在本論文中，我們使用套件 Museval[2]中的 bss_eval 函式來計算 SDR 值。我們遵照多數研究的設定，定義一首歌的 SDR 計算方式為將輸入歌曲的伴奏訊號與預測訊號切成一秒鐘的不重複片段來計算一秒的 SDR 值，隨後取所有片段的中位數當作一首歌的 SDR 值；最後，我們計算測試資料集中每首歌 SDR 的平均值及中位數當作整體結果，其中中位數是較為常見的設定，因此為相對重要的評估指標。

[2]https://github.com/sigsep/sigsep-mus-eval

doi:10.6342/NTU202303861

4.2.2 延遲時間

4.2.2.1 Test chunk

我們定義 test chunk 為測試時一次輸入模型的歌曲强度時頻譜片段。在音樂聲源分離任務中，多數模型在 test chunk 縮短時，分離品質會隨之下降。其原因有二，首先，在 U-Net 中的 CNN 架構，在進行輸入之前，通常會對特徵的邊界補 0，以避免輸入與輸出大小不相同的問題，但是這種操作在 test chunk 縮短時會造成分離時補 0 的區域比例增加，進而導致分離品質下降。再來，在使用時序架構的模型進行分離時，由於觀察視野在 test chunk 縮短時隨之減少，因此也會造成分離品質下降。在固定模型執行時間時，test chunk 越短，代表延遲越低。

4.2.2.2 RTF

RTF 的計算方式為輸入的處理時間與輸入時間長度的比值。在音樂聲源分離任務中，我們將一首歌的 RTF 定義為下列等式:

$$\text{RTF} := \frac{\text{Total processing time}}{\text{Total duration of a song}} \tag{4.3}$$

等式中的分子代表程式從一開始得到歌曲到最後輸出完整分離結果的時間，分母則是一首歌的實際長度。在固定歌曲長度時，模型處理時間越短，代表 RTF 越低，也代表延遲越低。通常 RTF<=1 代表能夠進行實時運算。

4.2.2.3 延遲時間計算

首先，由於我們的目標是即時分離，因此必須確保 RTF≤1。接著，我們就可以藉由 test chunk 及 RTF 來計算模型能夠達到的最佳延遲時間，假設 test chunk 長度為 T 秒，RTF 為 r，可得模型處理輸入的時間 Tr 秒，則最佳延遲時間 L 如以下等式:

$$L = 2Tr \tag{4.4}$$

推導過程請見附錄 A。所有實驗的延遲時間數值都是實際計算 test chunk 的時間長度與 RTF 後，使用上述等式求得。

doi:10.6342/NTU202303861

4.3 實驗設定

4.3.1 實驗環境

本次研究的實驗環境主要在兩台主機上進行，分別為實驗室主機及國高主機，設定分別如下:

- 實驗室主機
  - CPU: CPU Intel(R) Xeon(R) Silver 4116 @ 2.10GHz
  - RAM: 328 GB
  - GPU: NVIDIA Quadro GV100 and NVIDIA TITAN RTX
- 國高主機
  - CPU: Intel(R) Xeon(R) Gold 6154 @ 3.00GHz
  - RAM: 90 GB
  - GPU: NVIDIA Tesla V100 SXM2

此外，我們也有使用到邊緣裝置進行延遲的測試，由於裝置硬體資訊屬於機密，因此本論文並未列出。模型在訓練及測試時，皆使用單張 GPU 進行運算。進行 RTF 計算時，除了實驗五將模型套用於邊緣裝置外，我們皆將模型運行在實驗室主機，並限制 CPU 核心數為 1 以模擬資源稀少的情況。

4.3.2 實驗項目

- 實驗一: 模型訓練目標調整效果比較
  我們比較 3.3 中所述之不同訓練目標對 MMDenseNet 產生的影響，項目包含分離效果、參數量及 RTF。
- 實驗二: 模型架構調整效果比較
  我們比較 3.4 中所述之不同模型架構對 cIRM MMDenseNet 產生的影響，項目包含分離效果、參數量及 RTF。
- 實驗三: 對時間軸之自注意力機制架構消融實驗效果比較
  我們以 3.4.1 提出的架構當作消融實驗的基準模型，並比較內部架構的調整對分離效果及參數量產生的影響。

doi:10.6342/NTU202303861

- 實驗四: 漸進式測試與漸進式訓練效果比較
  我們以 3.4.1 提出的架構當作實驗四的基準模型，並比較 3.5 中提出的不同漸進式測試與漸進式訓練方法在縮短 test chunk 時的效果。
- 實驗五: 模型於邊緣裝置的表現分析
  我們將 MMDenseNet、HTDemucs、3.4.1 提出的架構、以 3.4.1 為架構的漸進式訓練模型放入邊緣裝置進行運算，並比較其分離效果、參數量、RTF 及 test chunk 長度。

4.3.3 訓練及測試設定

我們使用 MUSDB18 [12] 資料集中的歌曲進行訓練、驗證及分離品質測試，分別包含 86, 14, 50 首歌曲。本論文分離目標為伴奏聲源，做法為將資料集中歌曲的鼓聲音軌、貝斯音軌、及其他音軌進行加總。此外，本論文以八首平均為 243.25 秒的歌曲來計算 RTF 指標。

在本論文的研究中，模型輸入的特徵為歌曲由短時距傅立葉轉換 (Short-time Fourier Transform, STFT) 而得之強度時頻譜 $|X|$，以下是相關設定:

- FFT size: 2048
- Hop size: 1024
- Window function: Hann window function
- Sample rate: 44100 Hz
- Channels: 2 (stereo)

模型訓練的超參數設定如下:

- Batch size: 8 (Self-Attention, progressive training), 16 (other experiment)
- Loss: L1 loss on signal
- Epoch: 1300
- Optimizer: ADAM
- Learning rate: start from 1e-3, decay to 1e-5

doi:10.6342/NTU202303861

- Scheduler: StepLR (step size = 5, gamma = 0.99)

最後，在進行訓練及測試時，不同的方法會設定不同的資料片段長度，如表 4.1 所示。對於分離品質改進的方法，我們並不著重於延遲時間的縮短，因此盡量保持與基準模型的相同設定，模型架構調整實驗由於涉及自注意力機制的運算，因此特意增加 training chunk 的長度。延遲時間減少的實驗中，我們測試降低資料輸入長度對分離品質的改變，最低測試至 0.18 秒。

表 4.1: 訓練及測試資料片段長度實驗設定表

<table>
<tr><td rowspan="3">Methods</td><td>Training segment</td><td>Training chunk</td><td>Test chunk</td><td colspan="2">Look back chunk</td></tr>
<tr><td colspan="3"></td><td>Training</td><td>Testing</td></tr>
<tr><td colspan="5">(time frames, sec)</td></tr>
<tr><td>MMDenseNet</td><td>256, 5.94 sec</td><td>256, 5.94 sec</td><td>256, 5.94 sec</td><td colspan="2" rowspan="3">X</td></tr>
<tr><td>Learning target improvement</td><td>256, 5.94 sec</td><td>256, 5.94 sec</td><td>256, 5.94 sec</td></tr>
<tr><td>Model architecture improvement</td><td>512, 11.89 sec</td><td>512, 11.89 sec</td><td>256, 5.94 sec</td></tr>
<tr><td>Progressive inference</td><td>64, 1.48 sec</td><td>64, 1.48 sec</td><td>64, 1.48 sec</td><td>X</td><td>64, 1.48 sec</td></tr>
<tr><td rowspan="4">Progressive training</td><td rowspan="3">512, 11.89 sec</td><td rowspan="3">64, 1.48 sec</td><td rowspan="3">64, 1.48 sec</td><td>64, 1.48 sec</td><td rowspan="2">64, 1.48 sec</td></tr>
<tr><td rowspan="2">256, 5.94 sec</td></tr>
<tr><td>256, 5.94 sec</td></tr>
<tr><td>256, 5.94 sec</td><td>32, 0.74 sec</td><td>32, 0.74 sec</td><td>128, 2.97 sec</td><td>128, 2.97 sec</td></tr>
<tr><td>128, 2.97 sec</td><td>8, 0.18 sec</td><td>8, 0.18 sec</td><td>64, 1.48 sec</td><td>64, 1.48 sec</td></tr>
</table>

doi:10.6342/NTU202303861

第五章　實驗結果與討論

本章針對第 4 章中提到的五項實驗進行結果討論。我們針對分離效果、延遲時間、空間資源三個指標進行分析，並在最後挑選出本論文認為最適合於邊緣裝置進行實時分離的模型。

5.1　實驗一: 模型訓練目標調整效果比較

在這個實驗當中，我們比較 3.3 中調整訓練目標對模型表現的影響。我們自行訓練 MMDenseNet 作為基準模型。除此之外，我們依照原論文做法，將前述 MMDenseNet 模型套用 Wiener filter 當作後處理的方法，在此方法中，除了需要伴奏分離的模型外，也需要人聲分離的模型來進行 Wiener filter 的計算。本實驗將這兩種做法與訓練目標調整的實驗結果進行比較。我們於此實驗固定 test chunk 長度為 5.94 秒，表 5.1 為訓練目標調整的綜合評量結果。

5.1.1　分離品質比較

如圖 5.1 所示，MMDenseNet 的 median SDR 及 mean SDR 皆落在 11 左右，對應到主觀聽感來說殘存的人聲尚多，伴奏分離的表現並不突出。套用 Wiener filter 後，SDR 指標皆有大幅提升，主觀聽感上雖然仍會聽到一些殘存人聲，但可以明顯感受到相較於添加 Wiener filter 前已經少了許多。以强度遮罩 (magnitude mask) 為訓練目標的 MMDenseNet 模型 SDR 指標也有不錯的提升，消除的人聲效果大約比 Wiener filter 稍差一些。cIRM MMDenseNet 的 SDR 指標為這部分表現最好的模型，主觀聽感上與 Wiener filter 版本相比，在同首歌的不同片段互有優劣，然而 cIRM MMDenseNet 在消除人聲時，偶爾會出現高頻刺耳噪音，聽感舒適度上是 Wiener filter 版本表現較為自然。

doi:10.6342/NTU202303861

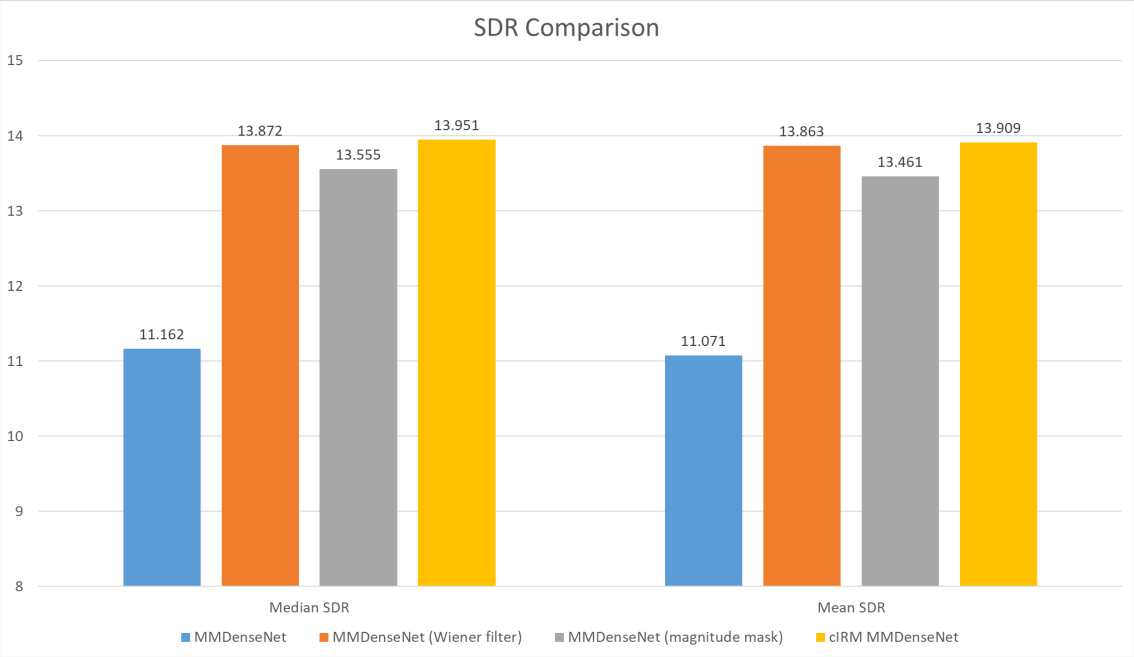


圖 5.1: 訓練目標調整與基準模型 SDR 比較圖

5.1.2 延遲時間比較

在這個實驗當中，由於我們固定 test chunk 的秒數，因此影響延遲時間的唯一因素為 RTF，圖 5.2 (a) 為訓練目標調整的延遲時間評量結果。首先比較 MMDenseNet 在添加 Wiener filter 前後的結果，可以發現使用 Wiener filter 後，延遲時間上升超過兩倍，且 RTF 接近實時運算的極限值，原因有二。首先，進行 Wiener filter 分離時需要運算伴奏分離及人聲分離兩個模型，模型運算量變成原模型的兩倍；再者，Wiener filter 的運算涉及 EM 演算法，需要進行許多輪的迭帶運算，這也會讓延遲時間上升，此結果也顯示 Wiener filter 不適合使用於本論文的研究。進行模型訓練目標調整後，由於並未調整模型整體架構，因此 RTF 指標皆維持在 0.4 以下，並無太大的改變，延遲時間不會增加太多。

5.1.3 空間資源比較

參數量評量結果如 5.2 (b) 所示。首先 MMDenseNet 在添加 Wiener filter 後，需要兩個模型的結果進行運算，所以參數量為原模型的兩倍。調整訓練目標為 madnitude mask 並不會影響模型參數量。cIRM MMDenseNet 則是因為有四項目標要預測，所以在神經網路最後的輸出層參數量會變為四倍，其餘架構則是不變，參數量僅是微幅上升。此實驗的模型大小都在 1M 以下，不會占用過多空間資源。

doi:10.6342/NTU202303861

(a) Latency comparison (b) Parameters comparison

圖 5.2: (a) : 訓練目標調整與基準模型延遲時間比較圖。(b) : 訓練目標調整與基準模型參數量比較圖。

表 5.1: 模型訓練目標調整與基準模型綜合比較表

Model architecture	Median SDR	Mean SDR	Latency (sec)	RTF	Parameters (K)
MMDenseNet	11.162	11.071	**4.38**	**0.3683**	**339**
MMDenseNet (Wiener filter)	13.872	13.863	9.99	0.8410	679
MMDenseNet (magnitude mask)	13.555	13.461	4.68	0.3943	**339**
cIRM MMDenseNet	**13.951**	**13.909**	4.39	0.3699	343

5.2 實驗二: 模型架構調整效果比較

在這個實驗當中，我們比較 3.4 中不同架構對模型表現的影響。我們使用 cIRM MMDenseNet 作為此實驗的基準模型。此實驗固定 test chunk 長度為 5.94 秒，表 5.2 為不同架構的綜合比較結果。

5.2.1 分離品質比較

分離品質評量結果如圖 5.3 所示。在對基準模型添加對時間軸的自注意力機制 (3.4.1) 後，mean SDR 及 median SDR 皆有所上升，客觀指標上數值表現提升許多；在主觀的聽感方面，分離出的伴奏相較基準模型有兩項改進。首先，此架構產生的伴奏殘留了較少的主唱人聲，再者，此方法消除了原先基準模型的高頻刺耳聲，基本上已經是合格的伴奏分離模型。基於 3.4.1 架構，再對 MDenseNet 使用軸向注意力機制後，median SDR 提升至 15、mean SDR 甚至突破了 17；主觀聽感方面，人聲已經被大部分消除，為一個優秀的伴奏分離模型。New cIRM MMDenseNet 為 3.4.1 的延伸架構，其 median SDR 及 mean SDR 皆有些微上升；主觀聽感上則是差異不大。此實驗印證了自注意力觀察視野的提升對分離品質的重要性，不論是對時間軸或是對頻率軸實行自注意力都對分離品質有所提升。

doi:10.6342/NTU202303861

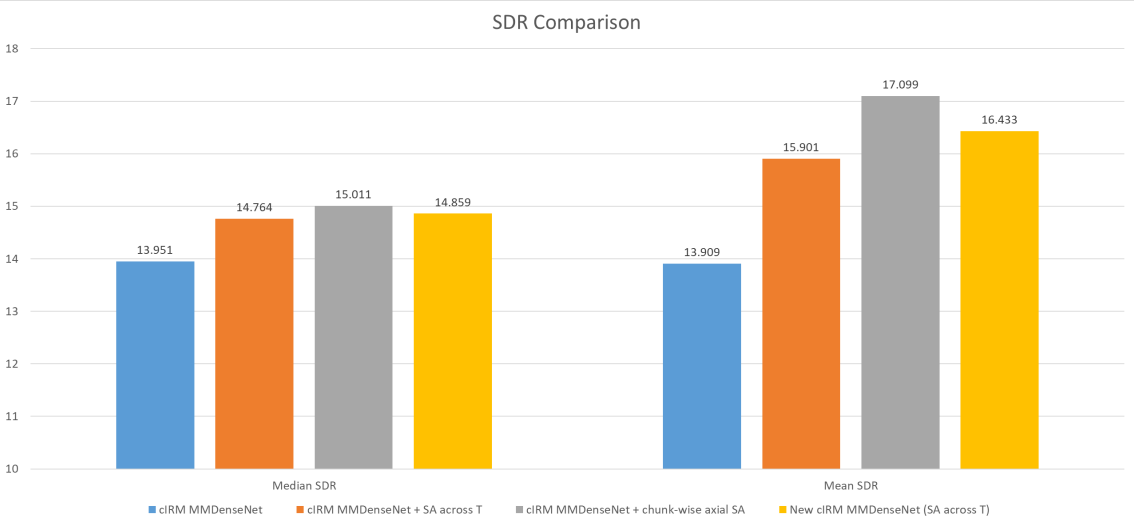


圖 5.3: 模型架構調整與基準模型 SDR 比較圖

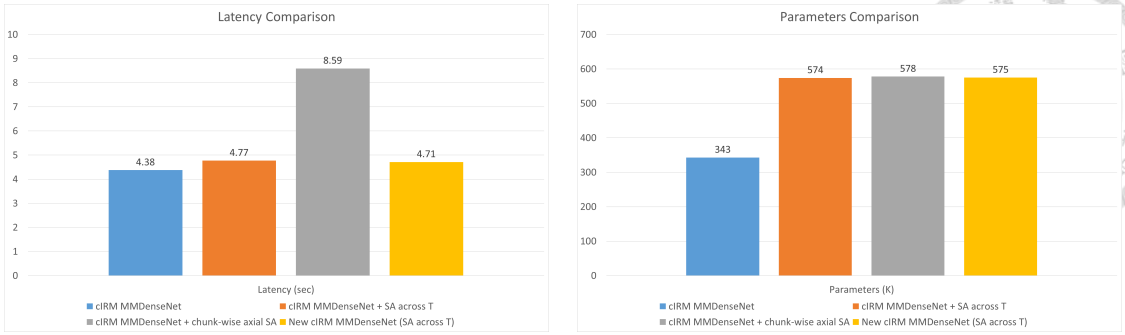
5.2.2 延遲時間比較

在這個實驗當中，由於我們固定 test chunk 的秒數，因此影響延遲時間的唯一因素為 RTF，圖 5.4 (a) 為不同架構的延遲時間比較結果。所有架構都是由 cIRM MMDenseNet 延伸而來，所以其延遲時間表現最好。在添加對時間軸的自注意力機制後，RTF 並沒有明顯增加，在延遲表現上與基準模型平分秋色。使用軸向注意力機制後，可以發現延遲時間增加許多，由於此方法對 full band MDenseNet 的幾乎所有 dense block 後皆進行了軸向注意力，這可能導致自注意力的運算量偏高，進而影響 RTF 表現。New cIRM MMDenseNet 架構在 RTF 的表現則是不錯，延遲時間與 cIRM MMDenseNet 差不多，可能的原因是此架構只有在特徵數量較少且較為精華的 bottleneck layer 使用軸向注意力，所需的運算量也就比較少。

5.2.3 空間資源比較

圖 5.4 (b) 為不同架構的參數量比較，由於此實驗中，所有架構都是由 cIRM MMDenseNet 延伸而來，所以其參數量最低。架構改良的三種方法參數量都在 1M 之下，且為原先架構的不到兩倍，這樣的增加幅度完全能夠接受，放進大多數邊緣裝置中也不會產生空間資源不足的問題。

doi:10.6342/NTU202303861



(a) Latency comparison (b) Parameters comparison

圖 5.4: (a)：模型架構調整與基準模型 RTF 比較圖。(b)：模型架構調整與基準模型參數量比較圖。

表 5.2: 模型架構調整與基準模型綜合比較表

Model architecture	median SDR	mean SDR	Latency (sec)	RTF	Parameters (K)
cIRM MMDenseNet	13.951	13.909	**4.38**	**0.3683**	**343**
cIRM MMDenseNet + SA across T	14.764	15.901	4.77	0.4016	574
cIRM MMDenseNet + chunk-wise axial SA	**15.011**	**17.099**	8.59	0.7233	578
New cIRM MMDenseNet (SA across T)	14.859	16.433	4.71	0.3968	575

5.3 實驗三: 對時間軸之自注意力機制架構消融實驗效果比較

此實驗針對對時間軸之自注意力機制中的不同架構進行消融實驗，並討論不同的調整對模型表現的影響。此實驗固定 test chunk 長度為 5.94 秒，表 5.3 為綜合評估結果。

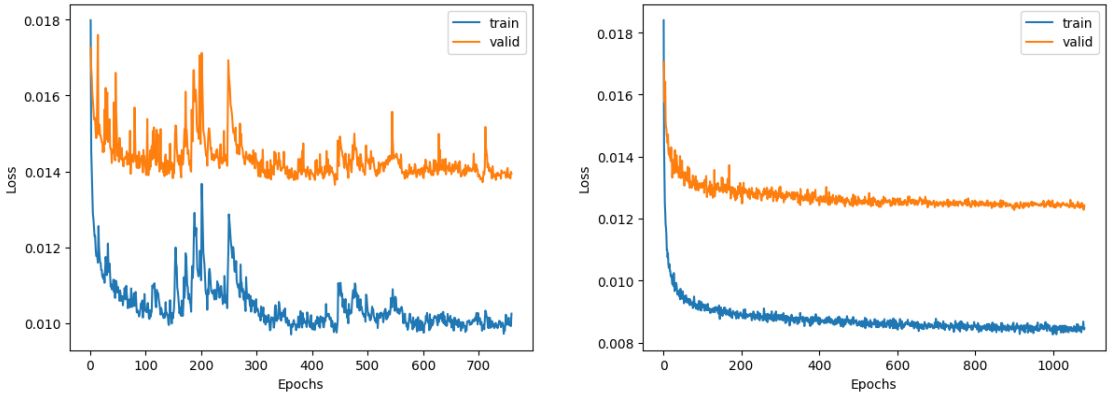
首先，我們比較使用線性層對模型表現所造成的影響，我們將線性層使用於產生 query 與 key 之前來達到抽象化及特徵降維的效果。分離品質方面，評量結果顯示，當我們不使用線性層之後，median SDR 及 mean SDR 皆下降了約 0.5，聽感上也有明顯的落差，這樣的結果能夠印證添加線性層對伴奏分離品質的重要性。觀察 RTF 的變化，可以發現是否添加線性層對於模型的延遲時間不會產生太大的變化。空間資源方面，拔除線性層大約會使參數減少 37 個百分點，與 cIRM MMDenseNet 的參數量差不多，不過原先的對時間軸之自注意力模型，其參數量就已經非常輕量，因此在現實層面，不至於有資源不足的問題。綜合以上結果，我們認為使用線性層對模型的影響利大於弊。

再來，我們將提出的多頭自注意力機制替換回原先 Liu [22] 所使用的單頭自注意力機制。分離品質方面，實驗結果顯示，使用多頭自注意力機制後，median SDR 指標能夠有效上升，mean SDR 也有些微的增加，對於提升伴奏分離品質有

doi:10.6342/NTU202303861

不錯的效果。在延遲時間方面，使用不同的自注意力機制於相同 test chunk 長度下，幾乎不會造成 RTF 的改變。兩種自注意力機制在參數量指標上有顯著的差距，使用提出的多頭注意力機制可以有效減少參數量，最大的原因是自注意力中線性層需要的參數量遠高於卷積神經網路，而替換成 C' 個頭自注意力會讓線性層的輸入減少為原先的 $1/C'$ 倍，在我們的實驗設定中，C' 為 5，這導致兩種設置的參數量差距十分明顯。基於以上評量結果，本論文提出的多頭自注意力機制在許多方面都優於原作法。

最後，我們討論 layer normalization 對模型架構的重要性。由於是否添加 layer normalization 不太會改變 RTF 及參數量，此部份我們專注於模型分離品質的討論。由結果可以發現，layer normalization 對 SDR 指標的影響相當巨大，移除 layer normalization 後，效果甚至不如未添加自注意力機制的 cIRM MMDenseNet。我們發現在訓練時，若未添加 layer normalization 於自注意力機制前，會導致模型的訓練變得很不穩定。添加 layer normalization 後，訓練穩定度明顯提升，且訓練損失及驗證損失皆能降至更低。圖 5.12 為兩種做法的比較結果。



(a) Epoch v.s L1 loss (w/o prenorm)　　(b) Epoch v.s L1 loss (w/ prenorm)

圖 5.5: (a)：添加 layer normalization 前訓練步數對模型訓練之損失圖。(b)：添加 layer normalization 後訓練步數對模型訓練之損失圖。

表 5.3: 自注意力機制消融實驗比較表。prenorm 代表的是否在自注意力機制前添加 layer normalization，linear layer 表示是在進行自注意力機制前添加線性層，multihead 為是否替換原自注意力機制為多頭自注意力機制。

Model architecture	Median SDR	Mean SDR	Latency (sec)	RTF	Parameters (K)
cIRM MMDenseNet	13.951	13.909	**4.39**	**0.3699**	**343**
cIRM MMDenseNet + SA across T	**14.764**	**15.901**	4.77	0.4016	574
- linear layer	14.203	15.421	4.76	0.4005	361
- multihead	14.457	15.898	4.75	0.3998	1,437
- multihead - prenorm	13.166	13.776	4.60	0.3873	1,426

doi:10.6342/NTU202303861

5.4 實驗四: 漸進式測試與漸進式訓練效果比較

本實驗對 3.5 所提出的延遲改進方法進行評量分析。在此實驗中，我們都是基於 3.4.1 所提出的對 cIRM MMDenseNet 進行時間軸的自注意力架構進行討論，本節之後所提到的基準模型都是指此模型架構。

在進行結果分析前，我們先觀察基準模型在不同長度的 training chunk 與 test chunk 時對分離表現的影響，結果如表 5.4 所示。首先，我們固定 training chunk 並測試在不同 test chunk 長度下的 median SDR 表現，從結果不難發現，當我們降低測試時的輸入長度時，分離品質下降得非常明顯。由此可知，我們的模型非常依賴自注意力機制所提供的全局資訊，當限縮其觀察視野到兩秒以下時 median SDR 已經低於 14，無法維持不錯的分離品質。再來，我們比較不同 training chunk 在相同 test chunk 長度下對 median SDR 的影響，當我們將 training chunk 降為與 test chunk 同為 1.48 秒時，發現 median SDR 有顯著的提升，這代表模型在訓練時無法觀看較長的片段，反而更專注於短秒數間的資訊，進而導致分離品質上升。於是我們認為當 training chunk 與 test chunk 長度相同時會有最好的分離效果，接下來的實驗都將套用這項設定。

表 5.4: 使用不同 training chunk 與 test chunk 對 median SDR 的結果比較表

Model architecture	Training chunk (sec)	Test chunk (sec)	Median SDR
cIRM MMDenseNet + SA across T	11.89	5.94	14.764
		2.97	14.003
		1.48	12.776
	1.48	1.48	13.969

5.4.1 漸進式測試結果分析

由於漸進式訓練時間較長，因此我們嘗試不添加漸進式訓練，單純將漸進式測試套用於基準模型的情形，並評估在兩種 test chunk 長度下 median SDR 及延遲時間的表現。整體評量結果如表 5.5 所呈現。

首先，我們比較不同漸進式測試方法的分離表現。根據圖 5.6 可以發現，當使用 look-back through bottleneck 時，不管 test chunk 為 1.48 秒或是 0.74 秒，median SDR 都略遜於基準模型。而使用 look-back through encoder blocks 時，雖然在 test chunk 為 1.48 秒時，median SDR 基本上有微幅上升，但秒數降至 0.74 秒時，median SDR 卻有一些下降。漸進式測試使用的兩種方法，對整體聽感都影

doi:10.6342/NTU202303861

響不大。我們認為漸進式測試的分離效果之所以沒有幫助，最大的原因是訓練與測試的設定不同，U-Net 的跳躍連接架構在漸進式測試時無法處理特徵視野不同的問題，導致模型分離表現無法上升。

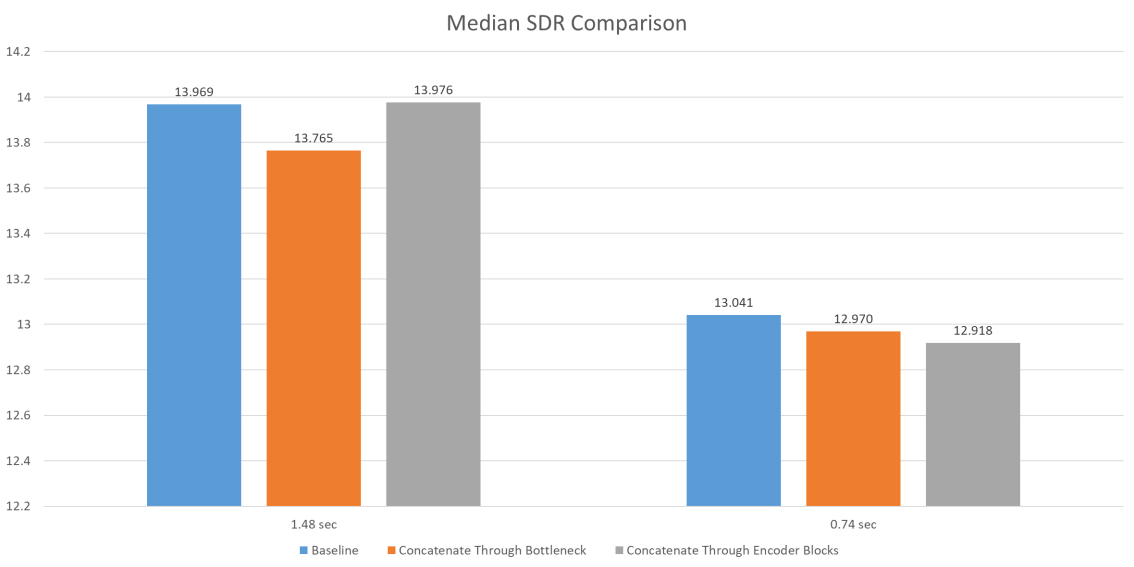


圖 5.6: 漸進式測試方法與基準模型 median SDR 比較圖

再來，我們比較不同方法的延遲時間評量結果。根據圖 5.7 可以發現，使用 look-back through bottleneck 不會讓模型增加額外的 RTF，單純改變 bottleneck layer 不會增加太多運算量，對延遲時間影響不大。套用 look-back through encoder blocks 時，由於在 decoder 的每一層皆需要額外的運算，所以會使 RTF 增加約 30%，實際運算時會增加約 0.4 秒的延遲時間。另外，我們發現將 test chunk 長度降低，會導致 RTF 略微上升，但在 0.5 秒以上差距都很細微。

表 5.5: 漸進式測試方法整體評量結果

Method	Test chunk (sec)	Look back chunk (sec)	Median SDR	Latency (sec)	RTF
Baseline	1.48	X	13.969	1.20	0.4068
	0.74	X	13.041	0.62	0.4177
Look-back through bottleneck	1.48	1.48	13.765	1.19	**0.4031**
	0.74	0.74	12.970	**0.61**	0.4090
Look-back through encoder blocks	1.48	1.48	**13.976**	1.58	0.5350
	0.74	0.74	12.918	0.83	0.5609

doi:10.6342/NTU202303861

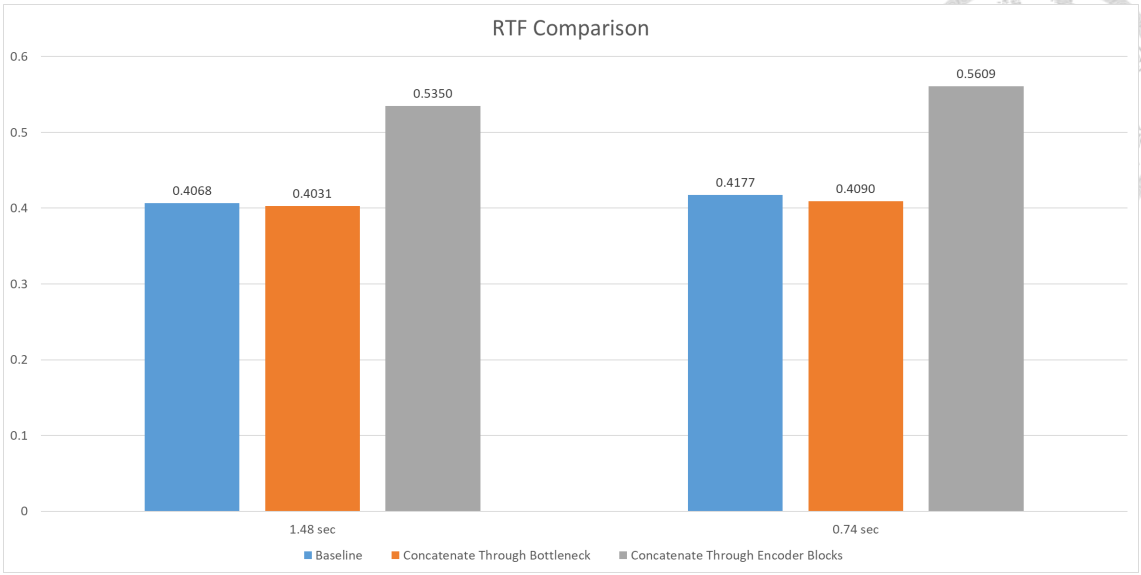


圖 5.7: 漸進式測試方法與基準模型 RTF 比較圖

5.4.2 漸進式訓練搭配漸進式測試結果分析

我們將漸進式訓練搭配漸進式測試套用於基準模型，並比較不同設定中 median SDR 及 RTF 的表現，使用的方法為 look-back through bottleneck。表 5.6 為漸進式訓練方法的所有評量結果。

首先，我們分析不同延遲時間下基準模型與漸進式訓練的結果。不同延遲時間的分離表現結果如圖 5.8 所示，我們以折線圖觀察 median SDR 對 latency 的趨勢。根據圖表可以很清楚的發現，在延遲時間降低時，分離品質呈現越來越差的趨勢。在使用漸進式訓練之後，在不同延遲情形下都能有效提高 median SDR。尤其當延遲時間縮短時，漸進式訓練的效果也會隨之提升。呈現的結果印證了漸進式測試在搭配漸進式訓練後確實能提升分離品質。

我們比較在固定 test chunk 下 RTF 的表現，因為固定 test chunk 時，RTF 與延遲時間成正比，結果如圖 5.9 所示。由於漸進式訓練搭配漸進式測試 look-back through bottleneck 的設定，所以可以視作與漸進式測試相同。本實驗將 test chunk 再度縮短為 0.18 秒，結果顯示，RTF 在此時開始上升，推判可能與 Pytorch 運行時的機制有關。

最後，我們發現訓練時 look back chunk 秒數越長，測試時分離品質越好，且差距十分明顯。另外，將訓練時較長 look back chunk 的模型於測試時縮短 look back chunk，雖然會造成分離品質的些微下降，但仍舊比一開始就訓練在較短 look back chunk 的模型表現好很多。

doi:10.6342/NTU202303861

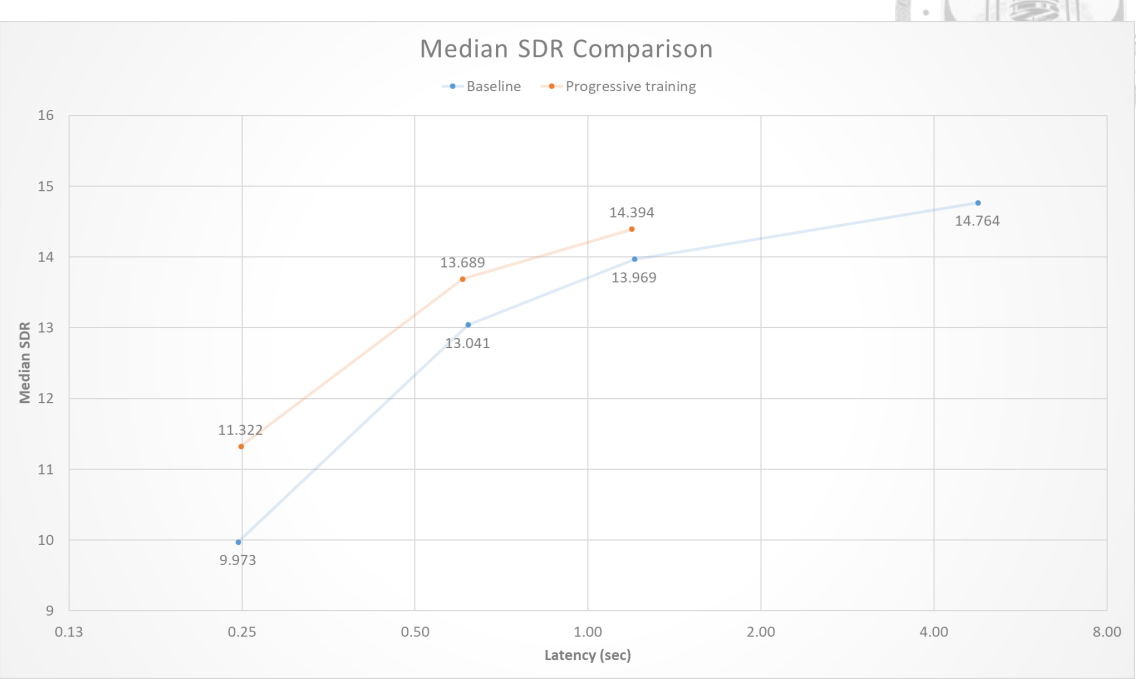


圖 5.8: 漸進式訓練方法與基準模型 median SDR 折線圖。由於 latency 最長的情況不符合漸進式訓練縮短延遲時間的目標，因此我們並未對此情形進行訓練。

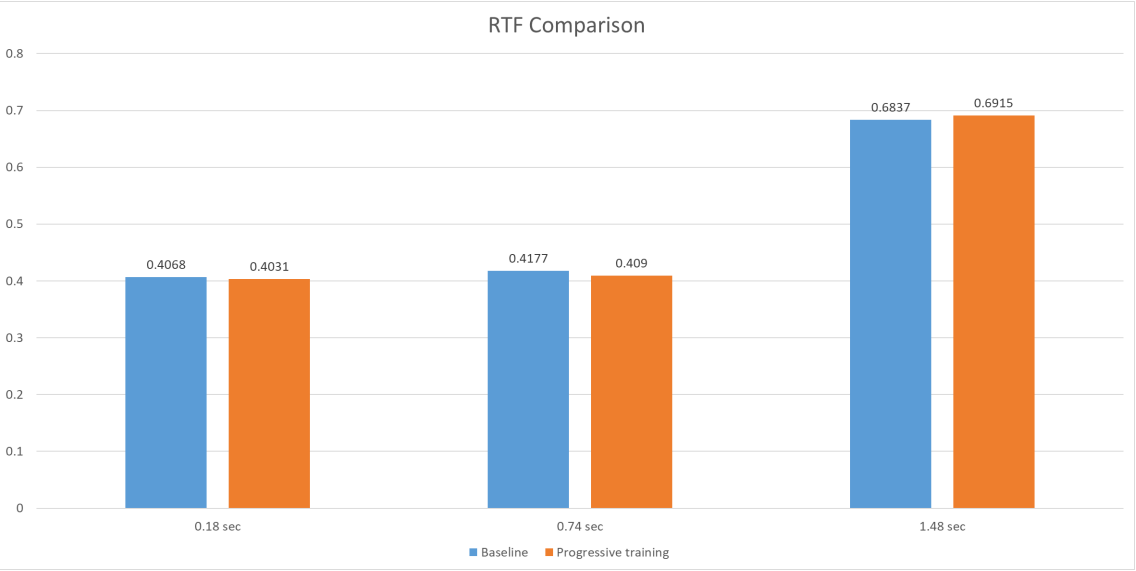


圖 5.9: 漸進式訓練方法與基準模型 RTF 比較圖

doi:10.6342/NTU202303861

表 5.6: 漸進式訓練搭配漸進式測試方法整體評量結果

Method	Test chunk / training chunk (sec)	Look back chunk (sec)		Median SDR	Latency (sec)	RTF
		Training	Testing			
Baseline	1.48	X		13.969	1.20	0.4068
	0.74			13.041	0.62	0.4177
	0.18			9.973	**0.25**	0.6837
Progressive training & progressive inference	1.48	1.48	1.48	14.048	1.19	0.4031
		5.94	0.74	14.327	1.13	**0.3812**
			1.48	14.350	1.16	0.3903
			5.94	**14.394**	1.19	0.4031
	0.74	2.97	2.97	13.689	0.61	0.4090
	0.18	1.48	1.48	11.322	**0.25**	0.6915

doi:10.6342/NTU202303861

5.5 實驗五: 模型於邊緣裝置的表現分析

此實驗將模型轉成 onnx 格式，並套用至邊緣裝置中進行表現評估，硬體限制將影響模型運行速度，因此主要會針對延遲時間的表現進行分析，並輔以模型的分離表現與空間資源來進行討論，如表 5.7 所示。在實驗中，我們使用的模型包含目前 state-of-the-art 模型 HTDemucs、MMDenseNet、使用漸進式訓練 + 測試前後的 cIRM MMDenseNet + 對時間軸的自注意力機制之模型。

首先，我們對模型分離表現進行討論。基於 mean SDR 與 median SDR 的指標表現，HTDemucs 會是我們最希望能夠於邊緣裝置進行實時分離的模型，其次是一般的 cIRM MMDenseNet + SA across T，再來是漸進式訓練 + 測試版本的模型，最後則是 MMDenseNet。前三名在 mean SDR 的差距不大，median SDR 則是 HTDemucs 最為突出，主觀聽感上 HTDemucs 伴奏品質最好，在大部分的歌曲片段完全聽不出殘留的人聲。

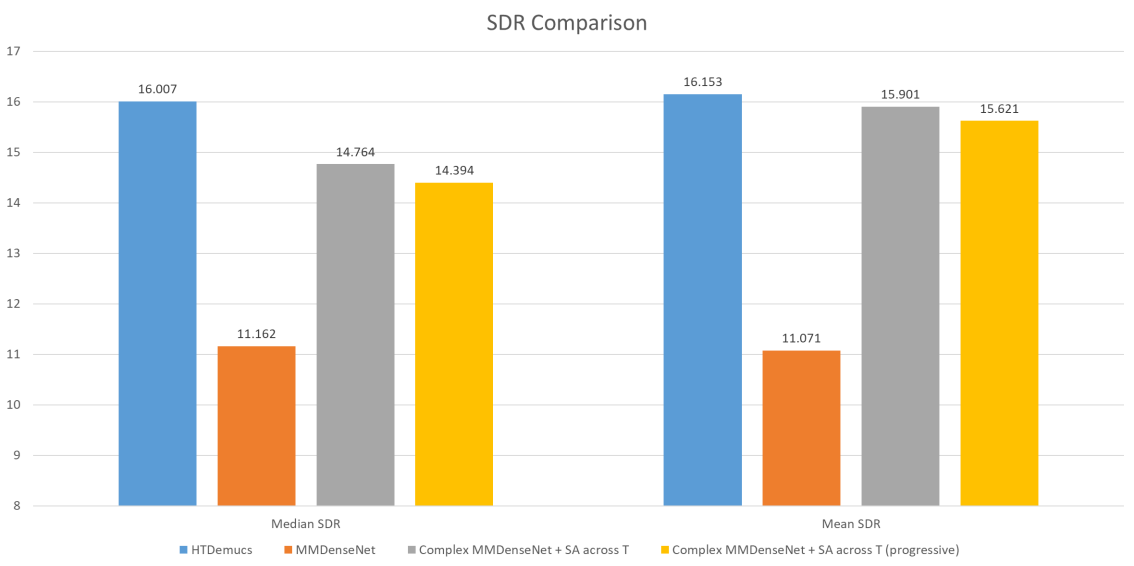


圖 5.10: 運行於邊緣裝置的模型 SDR 比較圖

再來，我們對模型參數量進行討論。從圖 5.11 可以發現除了 HTDemucs 外，其餘三個模型都能壓縮在 1M 以下，與 MMDenseNet 的差距更是接近百倍之多，在空間資源不夠大的嵌入式裝置中，會造成不小的負擔。反觀本論文提出基於 MMDenseNet 改進的作法，能夠在使用少量空間資源下進行分離。

最後，我們針對延遲時間進行討論，在此實驗中影響因素為 RTF 及 test chunk 長度。從分離品質的表現來看，我們優先希望 HTDemucs 能夠進行低延遲實時分離，然而當我們將 HTDemucs 模型送入邊緣裝置中後，發現其 RTF 指標

doi:10.6342/NTU202303861

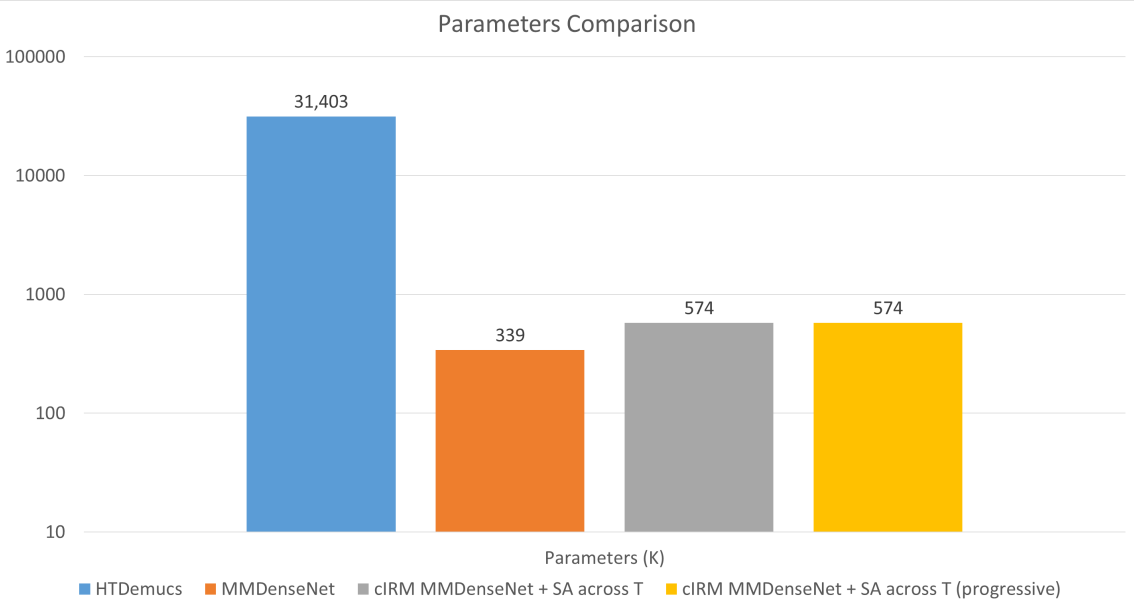
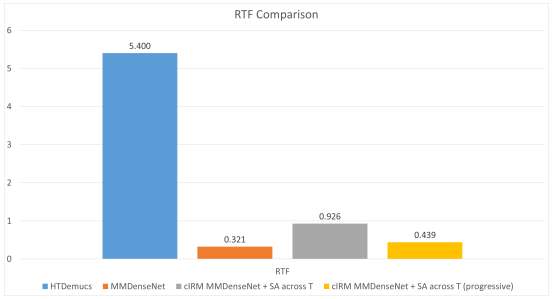
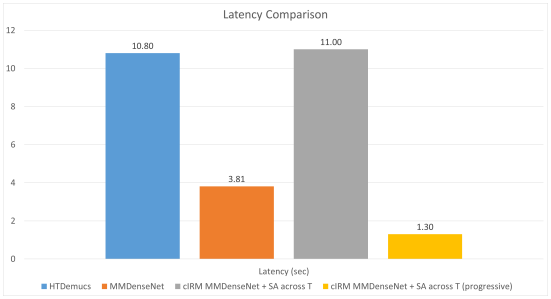


圖 5.11: 運行於邊緣裝置的模型參數量比較圖

過高導致無法進行實時分離，在一秒的 test chunk 下，裝置需要 5.4 秒來完成運算[1]。原 MMDenseNet 在設定當中，雖然 RTF 表現相對優秀，但是 test chunk 為 5.94 秒，造成延遲時間依舊過長，且分離品質並不突出。添加對時間軸的自注意力機制後，於邊緣裝置運行的 RTF 變成 0.926，加上 test chunk 長度不變，既使分離效果不錯，但是延遲時間卻比 MMDenseNet 還要長。對相同模型進行漸進式訓練與測試後，我們可以將 test chunk 縮短成原先的 25%，並且 RTF 也降低至 0.439，能夠做到接近一秒的延遲時間，且分離品質還能維持在不錯的水準[2]。此實驗結果印證了提出的自注意力機制與漸進式訓練及測試，確實能在三個目標中取得平衡，並成功應用於邊緣裝置。



(a) RTF comparison



(b) Latency comparison

圖 5.12: (a)：運行於邊緣裝置的模型 RTF 比較圖。(b)：運行於邊緣裝置的模型延遲時間比較圖。HTDemucs 模型的 RTF 大於 1，現實情況進行分離時，延遲時間應會比圖中計算值再更長。

[1]實際上於裝置進行分離時，雖然能夠運行，但是最後無法輸出正常的音檔。
[2]轉換成 onnx 格式後，median SDR 上升至 15.210，分離表現接近 HTDemucs，原因仍待釐清。

doi:10.6342/NTU202303861

表 5.7: 運行於邊緣裝置的模型綜合比較表

Model architecture	Median SDR	Mean SDR	Latency (sec)	RTF	Parameters (K)
HTDemucs	16.007	16.153	X	5.400	31,403
MMDenseNet	11.162	11.071	3.81	0.321	339
cIRM MMDenseNet + SA across T	14.764	15.901	11.00	0.926	574
cIRM MMDenseNet + SA across T (progressive)	14.394	15.621	1.30	0.439	574

5.6 綜合分析

我們將所有評量結果合併以進行分析，圖 5.13 為基準模型 MMDenseNet、分離表現的 SOTA 模型 HTDemucs 與基於 MMDenseNet 改進方法集合而成的泡泡圖。首先，在我們使用的方法中，分離表現最好的為 cIRM MMDenseNet 加軸向注意力的模型，其 median SDR 能超過 15。接著評估 RTF 的表現，可以發現在接近的延遲時間下，我們的改進方法除了 cIRM MMDenseNet 加軸向注意力的模型有些微增加外，其餘方法皆不太會增加模型處理時間，與基準模型的表現相去不遠，而 HTDemucs 則不符合實時分離的門檻。縮短延遲時間的情形下，我們發現漸進式訓練搭配漸進式測試能減緩分離品質的損失。最後，我們提出的方法將參數量維持在 1M 以下，為輕量化的伴奏分離模型。

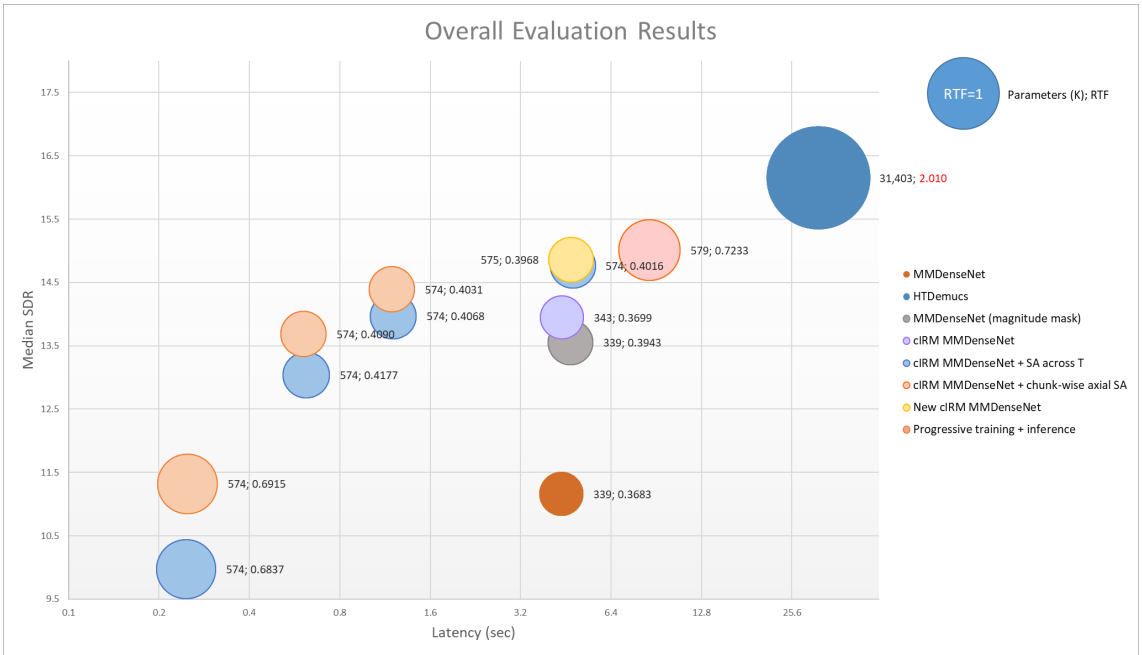


圖 5.13: 所有改進方法與基準模型綜合比較圖。泡泡大小為 RTF，右上角的泡泡為 RTF=1 時的大小，全部的實驗結果都是使用實驗室主機 CPU 進行測試。標籤值分號前後分別代表參數量及 RTF。HTDemucs 模型的 RTF 大於 1，現實情況進行分離時，延遲時間應會比圖中計算值再更長。

doi:10.6342/NTU202303861

第六章　總結

6.1　結論

本論文提出了一系列的改進方法，並以三個面向試圖改進輕量化模型 MMDenseNet，期望能達成伴奏的實時分離。此外，我們希望伴奏分離任務在三個目標中維持平衡，分別為低延遲、低空間資源、高分離品質。以下為本論文歸納出的重要結論:

1. 綜觀第 1、2 章所介紹的方法，我們發現音樂聲源分離的研究重心專注於提升分離品質，且近年來模型規模越來越大，這使模型難以應用於嵌入式裝置當中。另外，當模型縮短 test chunk 時，表現下降的非常明顯。我們發現 2017 年提出的 MMDenseNet 雖然分離表現不佳，但是有參數量少、RTF 低的特性，因此採用此模型當作基準模型，試圖提升分離品質並縮短 test chunk。

2. 由實驗一的評量結果可知，模型訓練目標的調整能夠有效提升 MMDenseNet 的分離品質，且不影響模型的參數量及 RTF。調整訓練目標為強度遮罩能夠限縮模型預測值域，進而提升分離品質。Complex MMDenseNet 基於 cIRM 分開預測強度及相位的資訊，為此實驗中表現最好的模型。我們採用此模型進行後續實驗。

3. 由實驗二的評量結果可知，調整模型架構能夠有效提升 Complex MMDenseNet 的分離品質。我們提出了三種不同方法，分別為對時間軸的自注意力架構、區塊式軸向自注意力機制、New Complex MMDenseNet。三種方法在 SDR 指標上都遠優於 Complex MMDenseNet，其中區塊式軸向自注意力機制的表現最為出色。此外，提出的方法中只有區塊式軸向自注意力機制會稍微增加 RTF，對延遲表現影響不大。最後，我們提出的方法會增加參數量，但是皆壓縮在 0.6 M 以下，表現仍舊出色。另外，我們於實驗三對自

doi:10.6342/NTU202303861

注意力機制中各個架構進行消融實驗，證明了我們提出的改進方法能有效提升 SDR 指標的表現，並大幅減少模型所需參數量。

4. 由實驗四的評量結果可知，我們提出的兩種漸進式測試方法直接套用於 Complex MMDenseNet + 對時間軸自注意力模型皆無法有效提升模型的分離品質。但是我們將其中一個方法，也就是 concatenate through bottleneck 搭配漸進式訓練後，發現在縮短 test chunk 的情形下，能夠有效減緩 median SDR 的下降，對於低延遲的分離品質有非常正向的效果。

5. 由實驗五的評量結果可知，將模型轉成 onnx 格式並送入邊緣裝置運算時，MMDenseNet 有分離品質低落的問題、HTDemucs 則是 RTF 過高無法進行即時運算。我們提出的模型架構在漸進式訓練的幫助下能夠在 1.48 秒的 test chunk 下，維持不錯的 RTF 及 SDR。這證明了本論文提出的模型改進及減少延遲方法能夠平衡上述提到的三項目標。

6.2 未來展望

雖然本論文提出了多種方法改進分離品質及降低延遲，但是由於時間及研究資源有限，其實仍有許多未被嘗試的方法可以進行探索，甚至是將本論文的研究做進一步的延伸。以下舉例多種延伸方向:

1. 音樂聲源分離任務中，資料的蒐集難度相較於其他任務較為困難，分軌的流行歌曲因為版權或其他因素變得難以取得，因此如何使用未經標注的資料就顯得格外重要。我們利用私人資料集，並使用 [9] 提出的方法進行半監督式訓練，然而結果都不盡理想。這部分仍有許多其他領域被提出的方法可以被嘗試，例如在電腦視覺領域非常有名的 self-training 方法 noisy student [31]、利用 consistency regularization 的 mean teacher [32] 等等。

2. 除了本論文使用的訓練目標調整方法，許多研究也提出將複數時頻譜直接當作模型的輸入，而非使用強度時頻譜，並預測複數遮罩。如音樂聲源分離模型 band-split RNN [9]、語音增強模型 FRCRN [33] 等。此外，為了讓模型更加輕量化，我們也可以將訓練目標換成更為簡單的特徵形式，如 [34] 所提出的 Mel-scaled mask。

3. 在我們的實驗設定中，只有將強度時頻譜平均切成高低兩個頻帶，然而高頻帶與低頻帶的重要性差異甚遠。根據表 6.1 所示，低頻帶的重要性遠超高頻

doi:10.6342/NTU202303861

帶，因此在未來的研究當中，可以在 New Complex MMDenseNet 的低頻帶處切更多的 subband，讓模型學習到較為重要的資訊，並於 bottleneck layer 共享，如此有機會增加模型的分離效果。

4. 從結論我們可以得知，漸進式訓練對漸進式測試方法 concatenate through bottleneck 的重要性，我們同樣也能將 concatenate through encoder blocks 方法套用於漸進式訓練當中。甚至我們可以嘗試各種不同的 look back 模式，例如讓模型跳過一些區塊來觀察更遠的視野，抑或是結合不同觀察視野的時序模型，例如套用 dual-path RNN [35] 的技術於自注意力機制的 SepFormer [36] 架構等等。

5. 對於分離表現提升的方法，除了論文中提及的 SDR 數值外，未來也可以嘗試進行歌曲的主觀分析，我們可以觀察加入漸進式訓練的方法後對不同曲風、不同頻率的分離表現來驗證方法的實用性。

6. 由於此論文的研究成果與瑞昱的研究計畫相關，計畫內容完全著重在分離伴奏聲源，因此目前的方法僅對伴奏的分離進行實驗。未來可以嘗試訓練不同聲源的分離模型，像是主唱、鼓聲等等，並比較不同聲源模型的分離表現。

表 6.1: 不同頻帶的損失比較表，評估模型為 Complex MMDenseNet 加上對時間軸之注意力機制。

Bands	Test chunk (sec)	Complex spectrogram loss (MSE)	Magnitude loss (MSE)	Phase loss (L1)	Average magnitude
Low frequency	1.48	0.882	1.186	0.617	1.194
	5.94	0.581	0.768	0.591	
High frequency	1.48	0.012	0.019	0.822	0.120
	5.94	0.005	0.009	0.745	

doi:10.6342/NTU202303861

doi:10.6342/NTU202303861

參考文獻

[1] N. Takahashi and Y. Mitsufuji, "Multi-scale multi-band densenets for audio source separation," in *2017 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, pp. 21–25, IEEE, 2017.

[2] S. Rouard, F. Massa, and A. Défossez, "Hybrid transformers for music source separation," *arXiv preprint arXiv:2211.08553*, 2022.

[3] D. Stoller, S. Ewert, and S. Dixon, "Wave-u-net: A multi-scale neural network for end-to-end audio source separation," *arXiv preprint arXiv:1806.03185*, 2018.

[4] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*, pp. 234–241, Springer, 2015.

[5] A. Défossez, N. Usunier, L. Bottou, and F. Bach, "Music source separation in the waveform domain," *arXiv preprint arXiv:1911.13254*, 2019.

[6] Z. Huang, W. Xu, and K. Yu, "Bidirectional lstm-crf models for sequence tagging," *arXiv preprint arXiv:1508.01991*, 2015.

[7] 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.

[8] W. Zhu and G. C. Beroza, "Phasenet: a deep-neural-network-based seismic arrival-time picking method," *Geophysical Journal International*, vol. 216, no. 1, pp. 261–273, 2019.

[9] Y. Luo and J. Yu, "Music source separation with band-split rnn," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 2023.

[10] 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.

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

[12] Z. Rafii, A. Liutkus, F.-R. Stöter, S. I. Mimilakis, and R. Bittner, "Musdb18-hq - an uncompressed version of musdb18," Aug. 2019.

doi:10.6342/NTU202303861

[13] 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 *Latent Variable Analysis and Signal Separation: 13th International Conference, LVA/ICA 2017, Grenoble, France, February 21-23, 2017, Proceedings 13*, pp. 323–332, Springer, 2017.

[14] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4700–4708, 2017.

[15] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

[16] Q. Kong, Y. Cao, H. Liu, K. Choi, and Y. Wang, "Decoupling magnitude and phase estimation with deep resunet for music source separation," *arXiv preprint arXiv:2109.05418*, 2021.

[17] D. S. Williamson, Y. Wang, and D. Wang, "Complex ratio masking for monaural speech separation," *IEEE/ACM transactions on audio, speech, and language processing*, vol. 24, no. 3, pp. 483–492, 2015.

[18] N. Takahashi, P. Agrawal, N. Goswami, and Y. Mitsufuji, "Phasenet: Discretized phase modeling with deep neural networks for audio source separation.," in *Interspeech*, pp. 2713–2717, 2018.

[19] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[20] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv:2010.11929*, 2020.

[21] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-end object detection with transformers," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part I 16*, pp. 213–229, Springer, 2020.

[22] Y. Liu, B. Thoshkahna, A. Milani, and T. Kristjansson, "Voice and accompaniment separation in music using self-attention convolutional neural network," *arXiv preprint arXiv:2003.08954*, 2020.

[23] D. Stoller, S. Ewert, and S. Dixon, "Adversarial semi-supervised audio source separation applied to singing voice extraction," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 2391–2395, IEEE, 2018.

[24] K. Li, X. Hu, and Y. Luo, "On the use of deep mask estimation module for neural source separation systems," *arXiv preprint arXiv:2206.07347*, 2022.

doi:10.6342/NTU202303861

[25] N. Wiener, N. Wiener, C. Mathematician, N. Wiener, N. Wiener, and C. Mathématicien, *Extrapolation, interpolation, and smoothing of stationary time series: with engineering applications*, vol. 113. MIT press Cambridge, MA, 1949.

[26] Z. Huang, X. Wang, L. Huang, C. Huang, Y. Wei, and W. Liu, "Ccnet: Criss-cross attention for semantic segmentation," in *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 603–612, 2019.

[27] H. Wang, Y. Zhu, B. Green, H. Adam, A. Yuille, and L.-C. Chen, "Axial-deeplab: Stand-alone axial-attention for panoptic segmentation," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part IV*, pp. 108–126, Springer, 2020.

[28] 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, pp. 155–160, 2014.

[29] 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, Guildford, UK, July 2–5, 2018, Proceedings 14*, pp. 293–305, Springer, 2018.

[30] 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.

[31] Q. Xie, M.-T. Luong, E. Hovy, and Q. V. Le, "Self-training with noisy student improves imagenet classification," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10687–10698, 2020.

[32] A. Tarvainen and H. Valpola, "Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results," *Advances in neural information processing systems*, vol. 30, 2017.

[33] S. Zhao, B. Ma, K. N. Watcharasupat, and W.-S. Gan, "Frcrn: Boosting feature representation using frequency recurrence for monaural speech enhancement," in *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 9281–9285, IEEE, 2022.

[34] M. Huber, G. Schindler, C. Schörkhuber, W. Roth, F. Pernkopf, and H. Fröning, "Towards real-time single-channel singing-voice separation with pruned multi-scaled densenets," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 806–810, IEEE, 2020.

[35] Y. Luo, Z. Chen, and T. Yoshioka, "Dual-path rnn: efficient long sequence modeling for time-domain single-channel speech separation," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 46–50, IEEE, 2020.

doi:10.6342/NTU202303861

[36] C. Subakan, M. Ravanelli, S. Cornell, M. Bronzi, and J. Zhong, "Attention is all you need in speech separation," in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 21–25, IEEE, 2021.

doi:10.6342/NTU202303861

附錄 A　最佳延遲時間推導

我們利用 RTF(real-time factor) 及 test chunk 長度來要進行最佳延遲時間的推導。首先，我們定義模型的 RTF 為 r，test chunk 長度為 T，則模型處理一個輸入的時間為 Tr。再者，由於我們的目標是進行即時分離，因此輸出必須保持連續，如此可以先排除 $r > 1$ 的情形。當 $r > 1$ 時，模型處理時間會大於 T，這會導致兩個模型輸出間產生間隔，如圖 A.1 所示。

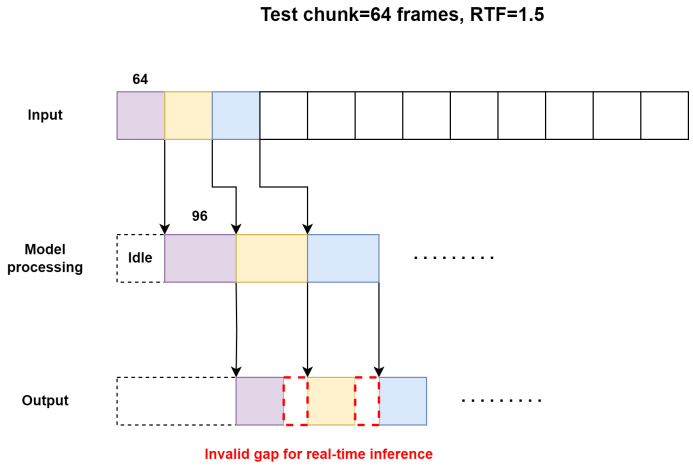


圖 A.1: 假設 test chunk 長度為 64 frames、RTF 為 1.5，則模型處理時間為 96 frames，大於輸入的 64 frames。這導致模型在串流時，無法即時輸出下一個時間段的結果。圖中紫色區塊的輸出結束後，模型仍未處理完黃色區塊，導致輸出時產生紅色虛線的間隔。

接著，我們就能夠分析 $r \leq 1$ 時，延遲時間最低的情況。由於模型處理時間為不可避免的延遲時間，因此關鍵在於何時能夠把輸入餵進去模型。在正常情況下，需要等到時間點 T 時才能組成一個 test chunk，因此延遲時間會是 test chunk 長度加上模型的處理時間，也就是 $T + Tr$，如圖 A.2 Situation 1 的情況。然而，當我們允許以下兩個設定時，就能縮短延遲時間:

1. 模型能夠接受經過 padding 後的輸入。
2. Test chunk 可以由過去模型處理過的輸入與尚未處理過的輸入所組成。

doi:10.6342/NTU202303861

在第一點的條件下，我們不需要等到輸入時間點 T 時才能組成一個 test chunk。這裡假設模型處理第一個輸入的時間點為第 t，並且會加上 $T-t$ 的 padding 以組成 test chunk 的大小。這表示我們能將延遲從 $T+Tr$ 減少至 $t+Tr$。考慮在這個情況下發生的改變:

1. 在處理第一個 test chunk 時，因為添加 padding 的關係，模型實際上有用的輸出長度會縮短成 t。

2. 模型能夠在 $t+Tr$ 時處理完第一個輸入並進行輸出，同時可以開始處理第二個 test chunk。第二個 test chunk 包含長度為 Tr 的尚未處理過的輸入，以及已經處理過的輸入。(若長度不足 T，則添加 padding。)

3. 基於前兩點，模型必須在開始第一個輸出後的 t 時間內處理完第二個輸出，否則將產生輸出間的空隙，導致輸出不連續。

由於模型處理時間固定為 Tr，因此如果要滿足上述第三點，必須讓 $t \geq Tr$，則模型的最佳延遲時間可以藉由以下公式得出:

$$\text{Optimal Latency} = t + Tr \geq Tr + Tr = 2Tr \tag{A.1}$$

而第二個以後的 test chunk 由於皆包含長度為 Tr 的尚未處理過的輸入，因此模型處理時間與輸出時間會相等，在最佳情況下能夠連續輸出。圖 A.2 Situation 2 為 optimal latency 的例子。

doi:10.6342/NTU202303861

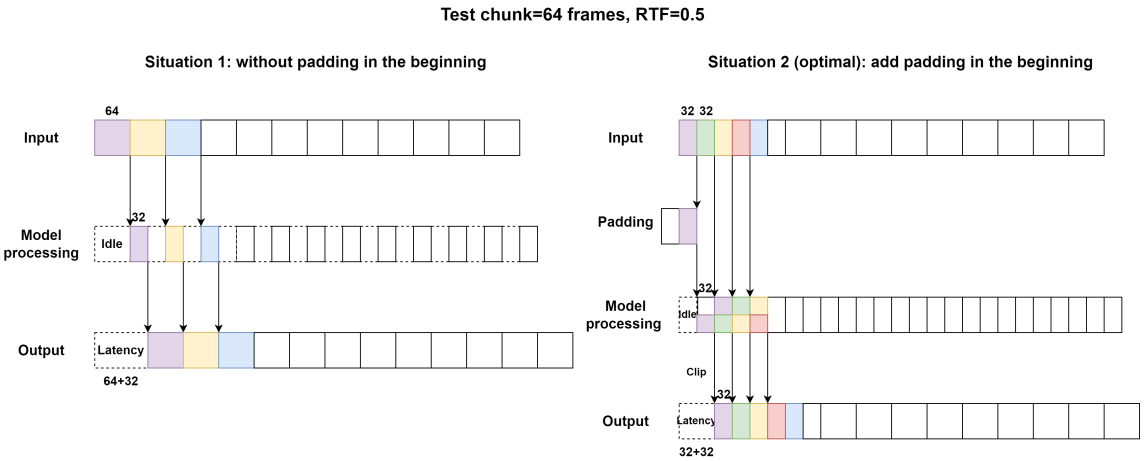


圖 A.2: 假設 test chunk 長度為 64 frames、RTF 為 0.5，則模型處理時間為 32 frames。Situation 1 為沒有添加 padding 的情況。Situation 2 時，由於添加了 padding，模型能於輸入 32 frames 時加上 padding 開始處理第一個 test chunk (白 + 紫)。於 64 frames 時，模型處理完第一個輸入，並開始輸出 (紫)，同一時間開始處理新的 32 個 frames 加上過去 32 個 frames (紫 + 綠)。於 96 frames 時，第一個輸出播放完畢，而模型剛好能夠及時處理完第二個輸入並開始輸出 (綠)，爾後以此類推。

doi:10.6342/NTU202303861