Un'Analisi Quantitativa e Visiva del Restauro Audio Zero-Shot con Demucs

August 29, 2025

Aurora Di Giovanna

Abstract

I modelli audio generativi producono spesso output con una qualità sonora inferiore agli standard professionali. L'obiettivo di questo progetto è l'implementazione di una pipeline di restauro che usi un approccio zero-shot, senza un addestramento specifico sul restauro, utilizzando un modello pre-addestrato per la separazione delle varie sorgenti musicali per migliorare gli audio.

1. Preparazione del dataset di valutazione

Per avere una valutazione oggettiva è stato creato un dataset di audio puliti e di alta qualità forniti dalla libreria *Librosa*. Gli audio forniti in formato *Ogg Vorbis* verranno, successivamente, convertiti in *WAV* per avere audio non compresso e ad altà fedeltà. Questo approccio è concorde ai metodi adottati nelle campagne di valutazione (10; 8).

1.1. Simulazione del degrado

In questa fase viene implementata una catena di degradazione. Per ogni audio pulito x, l'audio degradato $x_{deg} = F_{upsampling}(F_{noise}(F_{quantize}(F_{downsampling}(F_{lpf}(x)))))$ viene creato attraverso questa catena, formata da diverse funzioni:

- **Filtro Passa-Basso**: con la funzione *butter_lpf(x, sr, cutoff)* andiamo a creare un filtro che simula la perdita delle alte frequenze. La frequenza di taglio è a 7000 Hz e viene normalizzata rispetto alla sr/2, ovvero metà della frequenza di campionamento.
- **Downsampling**: il segnale viene ricampionato da 32kHz a 16kHz. Questo processo causa la perdita di informazioni (solitamente alte frequenze), in questo modo otteniamo degli audio più grezzi.
- **Quantizzazione a 8 bit**: *quantize_8bit_tpdf(x, rng)* andiamo a diminuire la risoluzione di ampiezza aggiungendo un dither (una distribuzione a forma triangolare), che aggiunge del rumore il quale modifica i livelli di ampiezza del segnale. Quando si riduce il numero di bit è necessario approssimare i valori usando la quantizzazione, che aggiunge errori.
- **Aggiunta del rumore**: con la funzione *add_noise_snr(x, snr_db, rng)* viene aggiunto del rumore che si adatta alla potenza del segnale di partenza. Questa relazione è descritta dal SNR Signal-to-Noise Ratio, ovvero una misura espressa in decibel, la quale confronta la potenza di un segnale con la potenza del rumore di fondo indesiderato. Quindi andiamo a calcolare per prima cosa la potenza del segnale: p1_signal = np.mean(x**2) + 1e-12, dove x è il nostro audio (che viene elevato al quadrato, poiché oscilla tra valori negativi e positivi). Viene poi calcolato il rumore in base alla potenza di segnale p_noise = p_signal / (10**(snr_db/10)) dopo di che viene sommato al segnale originale. Il rumore che viene creato viene generato da una distribuzione normale con media pari a zero in quanto non deve aggiungere distorsione e il segnale rimarrà centrato.
- **Upsampling**: infine il segnale viene riportato alla frequenza di campionamento originale di 32kHz usando un processo di upsampling tramite *resample_poly*. L'upsampling calcola i valori dei nuovi punti basandosi sui campioni vicini già esistenti usando l'interpolazione per aumentare la frequenza del segnale riportandolo al formato dell'originale per consentire un confronto. L'upsampling non recupera però l'informazione persa (le alte frequenze), il risultato è quindi un segnale tecnicamente a 32kHz, ma che contiene la stessa limitazione di un segnale a 16kHz, una perdita di dettaglio ad alta frequenza.

2. Pipeline di restauro

Per restaurare gli audio precedentemente degradati si è scelto di riconvertire un modello pre-addestrato per la separazione: **Demucs**, un modello di deep learning sviluppato da Meta AI introdotto in (3) e migliorato con un approccio ibrido in (2). Nonostante demucs sia stato creato per la separazione è stato studiato ed usato anche per il restauro audio, come nel denoising (9) e nel declipping (5). Questo

Email: Aurora Di Giovanna <digiovanna.2127738@studenti.uniroma1.it>.

Machine Learning 2025, Sapienza University of Rome, 2nd semester a.y. 2024/2025.

modello ha una struttura **Encoder-Decoder a U-Net** .

- **L'encoder**: l'encoder di Demucs processa l'audio in due modi paralleli:
  1. **Waveform**: il segnale di audio grezzo (forma d'onda) viene processato in diversi strati convoluzionali 1D;
  2. **Spectogram**: lo spettrogramma viene poi processato da strati convoluzionali 2D.

  L'encoder risponde alla domanda "Cosa c'è in questo audio?" prendendo lo spettrogramma degradato e processandolo in livelli, per ogni livello esegue due azioni: applica dei filtri per estrarre le caratteristiche (texture, forma...) e il downsampling finché non arriva al bottleneck, dove abbiamo una rappresentazione densa perché le dimensioni dei dati vengono ristrette.
- **Decoder**: il decoder di Demucs andrà a ricostruire l'intero segnale audio partendo dalla rappresentazione compressa.
- **U-Net**: L'architettura U-Net, introdotta originariamente per la segmentazione di immagini biomediche (6), è stata successivamente adattata con successo anche all'audio, ad esempio con Wave-U-Net (7). Questra struttura ha la forma di una U, con il processo di encoding andiamo ad eseguire la discesa e lentamente viene ridotta la risoluzione temporale e viene aumentata la profondità, ovvero il numero dei canali, costringendo il modello ad imparare suoni sempre più compressi. Il punto più profondo della "U" è il **bottleneck**, in questo livello Demucs inserisce degli strati a breve memoria che permettono ad esso di ricordare i pattern musicali. Con il processo di encoding andiamo ad eseguire la risalita aumentando la risoluzione temporale e in questa fase gli strati del percorso di discesa vengono connessi ai loro corrispondenti del percorso di risalita grazie alle skip connections, che permettono di combinare dati a bassa risoluzione con dettagli ad alta risoluzione.

3. Analisi Visiva e quantitativa

3.1. Analisi Spettrale

L'analisi spettrale fornisce una rappresentazione visiva del processo di restauro nel dominio della frequenza. Per ogni audio campione del dataset sono stati generati tre spettrogrammi in scala Mel: *Figure 1.* l'audio originale, *Figure 2.* l'audio degradato, il quale presenta una linea netta che elimina le alte frequenze, un risultato del filtro passa-basso e downsampling. In *Figure 3.* la versione restaurata dove notiamo un recupero delle alte frequenze molto simile all'originale, il che ci dimostra l'efficacia della pipeline di restauro costruita. In *Figure 4.* troviamo lo spettrogramma del residuo = s_{target} - $s_{restaurato}$, in cui notiamo che il modello ha avuto scarso successo nel gestire il rumore che si sovrappone alle frequenze basse del segnale audio ma è riuscito nel restaurare le alte frequenze.

Figure 1. Audio originale

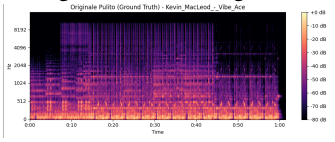


Figure 2. Audio degradato

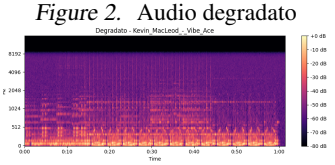


Figure 3. Audio restaurato

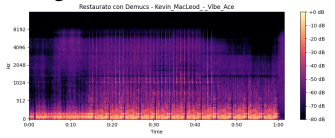
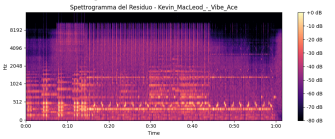


Figure 4. Residuo



3.2. Analisi quantitativa

Per l'analisi quantitativa sono state usate due metriche diverse: il Signal-to-Distortion Ratio (SDR) la cui formula è $SDR_{dB} = 10\log_{10}\left(\frac{||s_{target}||^2}{||s_{restaurato}-s_{target}||^2}\right)$ e lo Scale-Invariant SDR = $s_{target\_proj} = \frac{\langle \hat{s}_{restored}, s_{target} \rangle}{||\hat{s}_{restored}||^2}\hat{s}_{restored}$ (SI-SDR) (4) una misura più robusta e ormai standard nel campo degli audio.

Table 1. Audio di esempio "Kevin MacLeod - Vibe Ace".

Audio	SDR (dB)	SI-SDR (dB)
Degradato	-32.77	-64.06
Restaurato	-22.66	-29.63

Ulteriori esperimenti sono riportati nel notebook allegato.

4. Conclusioni

In conclusione, questo progetto ha dimostrato l'utilità di una pipeline di restauro zero-shot basata su Demucs, rivelando un miglioramento dell'SDR e al tempo stesso delle difficoltà nel restauro di audio complessi. In futuro potrà essere aggiunta un'analisi di ascolto formale come l'esperimento MUSHRA (1).

References

[1] Method for the subjective assessment of intermediate quality level of audio systems. Technical report, ITU-R, Rec. BS.1534-3, 2015.

[2] Alexandre Défossez. Hybrid spectrogram and waveform source separation. *arXiv preprint arXiv:2111.03600*, 2021.

[3] Alexandre Défossez, Nicolas Usunier, Léon Bottou, and Francis Bach. Music source separation in the waveform domain. *arXiv preprint arXiv:1911.13254*, 2019.

[4] Jonathan Le Roux et al.

[5] A. Marafioti et al. Audio declipping with deep neural networks. In *ICMLA*, 2019.

[6] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, pages 234–241. Springer, 2015.

[7] Daniel Stoller, Sebastian Ewert, and Simon Dixon. Wave-u-net: A multi-scale neural network for end-to-end audio source separation. In *19th International Society for Music Information Retrieval Conference (ISMIR)*, pages 334–340, 2018.

[8] Fabian-Robert Stöter et al. The 2018 signal separation evaluation campaign. In *LVA/ICA*, 2018.

[9] Cassia Valentini-Botinhao et al. Speech enhancement using deep recurrent neural networks. In *Interspeech*, 2016.

[10] Emmanuel Vincent et al. Performance measurement in blind audio source separation. *IEEE Transactions on Audio, Speech and Language Processing*, 2006.