



# Emotion Recognition from Audio using Deep Learning and Machine Learning Techniques

Giada Pietrocola, Paolo Simeone

Università degli Studi di Cassino e del Lazio Meridionale, Italia

## Introduzione

Il riconoscimento delle emozioni nel parlato è un compito fondamentale in vari ambiti, dalla salute mentale all’interazione uomo-macchina. Le emozioni vengono solitamente rappresentate tramite due dimensioni continue: valence (grado di positività) e arousal (livello di attivazione fisiologica). L’obiettivo di questo progetto è predire la valence a partire da segmenti audio, utilizzando sia approcci di deep learning che di machine learning classico.

Il dataset utilizzato è il MuSE dataset, fornito nell’ambito della MuSE Challenge. La pipeline include il preprocessing audio, la generazione di mel-spectrogrammi normalizzati, il training di un modello CNN-RNN con meccanismo di attention, l’estrazione di feature e l’utilizzo di regressori classici su tali feature.

- Training set: registrazioni con etichette complete, utilizzate per addestrare i modelli.
- Development set: file con suffisso \_devel, impiegati per tuning iperparametri e validazione.
- Test set: file con suffisso \_test, riservati alla valutazione finale.

Il dataset si distingue per la variabilità emotiva e per la difficoltà intrinseca dovuta alla spontaneità e al rumore presente nelle registrazioni. Poiché questo dataset faceva parte di una challenge, i file \_test non disponevano di etichette. Per ovviare a ciò, abbiamo deciso di impiegare i file \_train e \_devel forniti, come nostri set di test. Per tutti gli altri file disponibili, ciascuno della durata approssimativa di 5 minuti, abbiamo applicato una divisione 80-20, dedicando l’80% all’addestramento e il 20% al testing.

## Dataset

Il dataset utilizzato proviene dalla MuSE Challenge, un benchmark per il riconoscimento multimediale dello stress e dell’emozione. La componente audio, oggetto del nostro studio, include conversazioni spontanee registrate in ambienti realistici. Le annotazioni di valence e arousal sono fornite come serie temporali, ottenute da annotatori umani e poi allineate e mediate.

Ogni traccia audio è associata a due file CSV contenenti valori continui di valence e arousal annotati a intervalli di 500 ms. I dati sono segmentati come segue:

## Mel-spectrogrammi e Preprocessing

La rappresentazione audio scelta è il mel-spectrogramma, una trasformazione del segnale audio in dominio tempo-frequenza che simula la percezione umana dell’altezza sonora.

Il processo di preprocessing segue questi passi:

1. Segmentazione: ogni file audio viene suddiviso in finestre di 0.5 secondi (8000 campioni a 16 kHz).

2. Normalizzazione: ogni segmento è portato a un livello standard di -3 dBFS per garantire uniformità di volume.
3. Data augmentation: con probabilità 0.5, vengono applicate trasformazioni casuali:

Tecnica	Parametri
Rumore bianco	$\in [0.01, 0.05]$
Pitch Shift	$\in [-2, 2]$ semitonni
Variazioni di velocità	$\in [0.85x, 1.15x]$
Modifiche di volume	$\in [0.9, 1.1]$

4. Estrazione mel: vengono calcolati 128 coefficienti mel con librosa, usando una FFT di 1024 e hop length di 256.
5. Conversione logaritmica: i valori di potenza sono trasformati in dB.
6. Normalizzazione globale: vengono applicati mean e std calcolati sull'intero training set per standardizzare tutti i dati.

Ogni sequenza consiste di 40 segmenti consecutivi (20 secondi), e viene salvata come file .npz contenente lo spettrogramma (forma [40, 1, 128, 34]) e le etichette corrispondenti ([40, 2]).

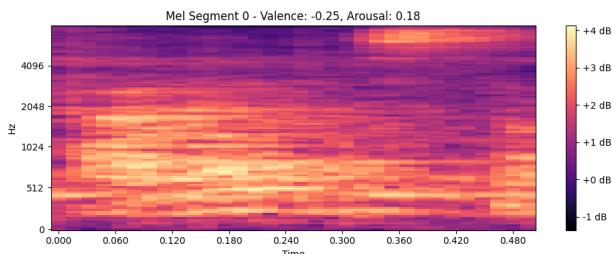


Figura 1: Mel-spectrogramma di una finestra di 500 ms

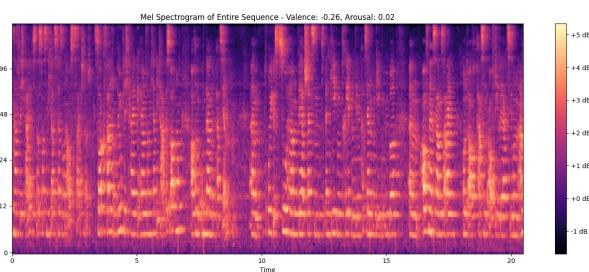


Figura 2: Mel-spectrogramma di una sequenza di 40 segmenti

## Deep Learning

### Modello

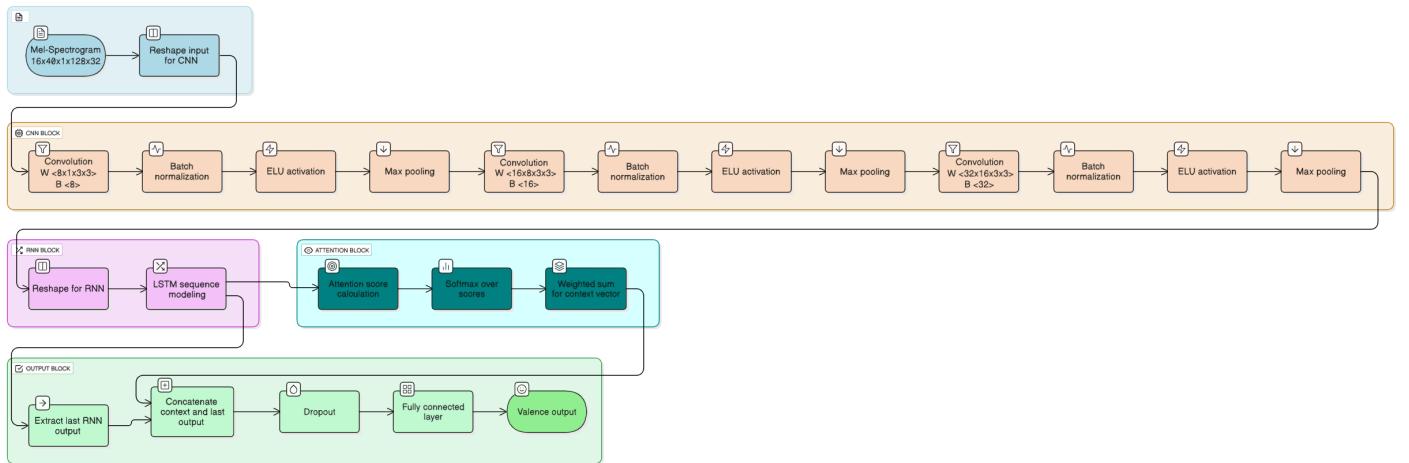
Il modello sviluppato appartiene alla categoria CNN-RNN con Attention, una combinazione efficace per dati strutturati in sequenza temporale e con contenuto spettrale, come i mel-spettrogrammi di segmenti vocali. Questa architettura è stata progettata per sfruttare contemporaneamente la capacità delle reti convoluzionali di estrarre caratteristiche spaziali dai singoli spettrogrammi e la capacità delle reti ricorrenti di modellare la dinamica temporale su sequenze estese.

**Blocco Convoluzionale (CNN)** Il blocco convoluzionale è composto da tre layer successivi, ciascuno dei quali esegue una convoluzione 2D seguita da Batch Normalization, funzione di attivazione ELU e Max Pooling. Le dimensioni spaziali si riducono progressivamente, mentre aumenta la profondità delle feature map (da 8 a 32 canali). In particolare:

- il primo livello estrae pattern di basso livello (es. variazioni locali di energia);
- i successivi livelli aggregano informazioni più astratte come andamenti tonali o strutture ritmiche ricorrenti.

**Blocco Ricorrente (LSTM)** Dopo la fase convoluzionale, le sequenze risultanti (una per ogni segmento) vengono appiattite e concatenate per formare una sequenza temporale tridimensionale. Questa viene processata da un LSTM bidirezionale a 2 layer, che consente di catturare relazioni temporali passate e future tra i segmenti. Questo è particolarmente utile nel contesto del parlato, dove l'intonazione e il ritmo sono essenziali per la comprensione del tono emotivo.

**Meccanismo di Attention** Una volta ottenute le hidden state per ogni timestep dalla LSTM, viene applicata un'operazione di attenzione. Un layer lineare produce pesi scalari per ciascun timestep, i quali vengono trasformati in una distribuzione di probabilità tramite softmax. La somma pesata delle hidden states rappresenta un contesto globale della sequenza, focalizzato sui momenti ritenuti più informativi.



**Figura 3:** Architettura della rete neurale proposta

**Concatenazione e Testata di Output** La rappresentazione finale è costituita dalla concatenazione del contesto pesato (attention output) con l'ultima hidden state. Questa è poi passata attraverso un layer fully-connected che produce un singolo valore continuo, corrispondente al valence. In questa versione il modello è stato focalizzato esclusivamente sulla valence, ma estendibile facilmente anche all'arousal.

La funzione obiettivo è la Concordance Correlation Coefficient (CCC), che combina correlazione lineare, deviazione standard e differenza tra medie. È preferita rispetto a MSE/MAE in quanto misura la coerenza globale tra le curve predetta e reale.

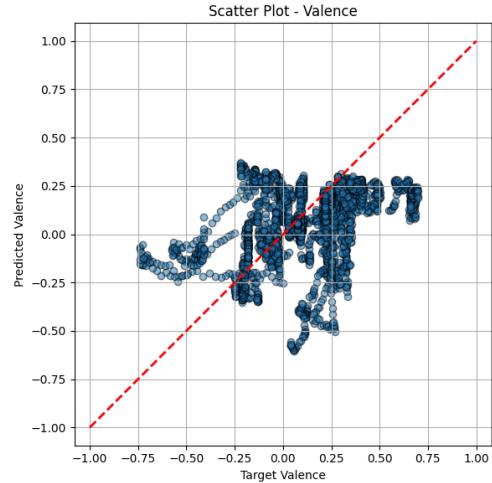
Il modello è stato addestrato su GPU con batch size pari a 64, ottimizzazione Adam con weight decay e learning rate scheduler Cosine Annealing per adattare dinamicamente la velocità di apprendimento. È stato applicato early stopping basato sulla CCC su validation.

## Risultati

Prestazioni del modello:

Metrica	Valore
CCC	0.3359
MSE	0.0802
MAE	0.2246

Questi risultati mostrano che il modello è in grado di apprendere una rappresentazione coe-



**Figura 4:** Scatter plot Valence: Valori predetti vs. Valori Target

rente dell'evoluzione emozionale, sebbene le previsioni risultino leggermente smorzate nei picchi. Questo comportamento è probabilmente dovuto sia all'effetto medio dell'LSTM sia alla variabilità soggettiva delle annotazioni umane.

Analizzando le predizioni in forma di scatter plot, si osserva una buona distribuzione lungo la diagonale, ma con una certa tendenza a concentrare i valori nella zona centrale, indicando una difficoltà nel modellare i casi più estremi (es. valence molto negativo o molto positivo).

## Problematiche

Durante lo sviluppo del progetto sono emerse diverse problematiche tecniche e sperimentali

che hanno richiesto l'introduzione di soluzioni specifiche.

**Checkpointing** Uno dei principali vincoli operativi è stato l'utilizzo di Google Colab come ambiente di addestramento. A causa dei limiti di sessione è stato necessario progettare un sistema robusto di salvataggio e ripristino dello stato di addestramento.

Per questo motivo è stato implementato un meccanismo di checkpointing personalizzato, che salva: epoca corrente, stato del modello (parametri dei layer), stato dell'ottimizzatore, stato dello scheduler del learning rate, cronologia delle metriche di training e validazione.

Questo ha permesso di interrompere l'addestramento su un account Colab e riprendere esattamente dalla stessa epoca e condizioni da un altro account o dispositivo.

**Problemi di apprendimento iniziali** Nelle prime versioni del modello, le prestazioni erano insoddisfacenti: il CCC risultava costantemente inferiore a 0.2, anche dopo diverse epoche di training.

Tra le ipotesi considerate:

- Redundancy nei dati: lo stride iniziale nella generazione delle sequenze era impostato a 1, producendo sequenze fortemente sovrapposte e quindi simili, ma associate a etichette leggermente diverse. Questo poteva introdurre ambiguità nel mapping tra input e target.
- Normalizzazione locale errata: inizialmente ogni mel-spectrogramma veniva normalizzato singolarmente tramite Min-Max scaling. Questo approccio, pur mantenendo valori compresi tra 0 e 1, annullava le differenze relative tra i segmenti e introduceva una distorsione dinamica nelle rappresentazioni spettrali.

Sono quindi state condotte due modifiche sperimentali:

- aumento dello stride da 1 a 10 per ridurre la ridondanza delle sequenze;
- sostituzione della normalizzazione per segmento con una normalizzazione globale, calcolata come media e deviazione standard su tutto il training set.

I risultati hanno mostrato chiaramente che:

la normalizzazione globale ha portato a un netto miglioramento del CCC rendendo le rappresentazioni molto più coerenti e comparabili tra segmenti diversi;

il ritorno allo stride=1, con la nuova normalizzazione, ha ulteriormente migliorato leggermente le prestazioni, suggerendo che la ridondanza non era la causa principale del problema iniziale.

## Machine Learning

Dopo l'addestramento del modello deep, è stata intrapresa una seconda fase con l'obiettivo di sfruttare le rappresentazioni apprese (deep features) come input per modelli di regressione classici. Questa strategia, nota come feature-based transfer learning, permette di ridurre drasticamente il costo computazionale della predizione, mantenendo parte delle performance del modello profondo.

Le feature sono ottenute dalla concatenazione dell'output del meccanismo di attention e dell'ultima hidden state del modello LSTM. Questo vettore di 512 dimensioni rappresenta una codifica sintetica dell'intera sequenza audio di 20 secondi.

Per esplorare le potenzialità di queste feature, è stata costruita una griglia di modelli e preprocessori:

**Scaler** Per garantire la compatibilità tra vari modelli e una distribuzione omogenea delle feature, sono stati testati:

Scaler	Descrizione
Nessuno	Identity Transformer
StandardScaler	Normalizzazione (media 0, varianza 1)
MinMaxScaler	Scalatura in range [0,1]]

**Riduttori di dimensionalità** Per valutare l'impatto di diverse tecniche di selezione e riduzione della dimensionalità, sono stati testati:

Tipologia	Parametri
Nessuno	Identity Transformer
SelectKBest	k = 16
PCA	n_components = 10
PCA	n_components = 20

## Modelli di regressione

- **Linear Regressor:** Modello di base che assume una relazione lineare tra feature e target;
- **Ridge Regressor:** estensione della regressione lineare con regolarizzazione L2, che penalizza i pesi troppo grandi;
- **SVR:** regressore non lineare basato su SVM, con kernel polinomiale (grado 5);
- **XGBoost:** regressore basato su boosting di alberi decisionali (200 alberi, max depth=1);
- **Ensemble:** combinazione del modello di regressione lineare e SVR.

Il processo ha generato 60 combinazioni diverse, ognuna delle quali è stata addestrata su training e valutata sul test set. Il criterio di selezione finale è stato il CCC, per garantire coerenza con il modello deep.

## Risultati

Le combinazioni testate mostrano un'ampia variabilità nelle prestazioni, con valori di CCC che spaziano da meno di 0.1 fino a 0.40. È interessante notare che i migliori modelli di regressione classici hanno superato il modello di deep learning, dimostrando l'efficacia delle feature estratte.

Scaler	Reducer	Model	CCC
none	knn	SVR	0.400
none	none	Linear	0.380
none	none	Ensemble	0.362
none	none	SVR	0.324
minmax	none	Ridge	0.325

Prestazioni delle 5 migliori combinazioni (Scaler, Riduttore, Modello) per CCC

## Preprocessing

- Le configurazioni senza scaler né riduzione (scaler="none", reducer="none") occupano i primi posti: le feature deep, normalizzate globalmente, sono già omogenee e ricche di informazione utile.
- L'uso di StandardScaler o MinMaxScaler non offre vantaggi sistematici e, in alcuni casi, degrada leggermente il CCC, probabilmente perché altera la distribuzione naturale dei dati deep.

- La PCA (10 o 20 componenti) riduce troppo la dimensionalità, perdendo dettagli emotivi sottili e portando CCC sotto 0.25.

## Modelli

- L'SVR con kernel polinomiale grado 5 si conferma top-performer quando lavora su feature non trasformate o leggermente selezionate: il SelectKBest (basato su mutua informazione) elimina la ridondanza senza intaccare l'informazione critica, portando il CCC fino a 0.40.
- La Linear Regression semplice raggiunge CCC = 0.38, segno che una parte significativa dell'informazione valence è già rappresentabile in modo lineare nello spazio delle feature deep.
- Ensemble (Voting): combina Linear Regression e SVR, ottenendo CCC = 0.36, inferiore ai rispettivi 0.38 e 0.40, perché il voting uniforme disperde i punti di forza di ciascun modello.
- Ridge Regression: con penalizzazione L2 comprime troppo i coefficienti, attenuando variazioni informative chiave e portando il CCC a circa 0.28, rispetto al 0.38 del modello lineare puro.
- XGBoost, pur essendo un metodo potente, raggiunge al massimo CCC = 0.24. Ciò suggerisce che, sulle feature deep impiegate, i modelli a boosting di alberi non riescono a catturare approfonditamente la struttura continua dell'emozione come fanno SVR e i modelli lineari.