Undercomplete Autoencoder per Analisi e Clustering di Spettri Raman

Undercomplete Autoencoder per Analisi e Clustering di Spettri Raman

Amelia Carolina Sparavigna[1] e **Gemini (Modello Linguistico di Google)**[2]

[1] DISAT, Politecnico di Torino, [2] Gemini AI

DOI: 10.5281/zenodo.18339887

L'analisi spettroscopica Raman di matrici carboniose complesse, come il biochar, è spesso limitata da un basso rapporto segnale/rumore e da intense interferenze di fluorescenza. In questo lavoro, presentiamo un approccio basato su un Undercomplete Autoencoder (AE) per l'estrazione non supervisionata di "pseudo-spettri". A differenza delle tecniche di filtraggio tradizionali, il modello utilizza un collo di bottiglia (bottleneck) a 32 dimensioni per operare una filtrazione intelligente, eliminando il rumore stocastico e ricostruendo la morfologia archetipica del segnale. Dimostriamo come lo pseudo-spettro \hat{x}, generato dalla decodifica dello spazio latente z, agisca come centroide visibile di cluster di materiali simili. L'errore di ricostruzione (MSE) viene inoltre proposto come sensore di anomalia per l'identificazione di contaminanti o variazioni strutturali non presenti nel dataset di addestramento.

1. Introduzione e Obiettivo

L'analisi Raman su materiali complessi (come il biochar o campioni contaminati da detergenti) è spesso inficiata da un basso rapporto segnale/rumore (S/N) e da forti fondi di fluorescenza. L'obiettivo di questo approccio è superare il denoising tradizionale (es. filtri Savitzky-Golay) utilizzando un Autoencoder (AE) per estrarre uno "pseudo-spettro" pulito e utilizzarlo come descrittore per il clustering.

2. Il Concetto di Pseudo-spettro e il Clustering Latente

2.1 Lo Pseudo-spettro: L'essenza oltre il rumore

Nel contesto di questo studio, definiamo **"Pseudo-spettro"** la proiezione dello spettro originale x ricostruita dal Decoder (\hat{x}) dopo essere passata attraverso il collo di bottiglia (*bottleneck*) dello spazio latente. A differenza di un semplice spettro filtrato matematicamente (come avviene con il filtro di Savitzky-Golay), lo pseudo-spettro è una **costruzione generativa**:

- **Filtrazione Intelligente:** Poiché il modello ha una capacità limitata nel *bottleneck* (solo 32 neuroni per 1024, ad esempio, punti di input), esso non può memorizzare il rumore casuale e stocastico, che per sua natura è privo di struttura correlata.
- **Ricostruzione Morfologica:** Il modello impara a dare priorità alle caratteristiche ricorrenti e fisicamente rilevanti, come i picchi Raman e la curvatura della fluorescenza.

- **Risultato:** Lo pseudo-spettro \hat{x} è quindi una versione "idealizzata" del segnale, che conserva l'informazione chimica (ad esempio, le bande D e G del biochar) eliminando le fluttuazioni che ne renderebbero difficile l'analisi quantitativa.

2.2 Il Clustering nello Spazio Latente: Confronto per Firma Digitale

Il **Clustering** è il processo di raggruppamento automatico di campioni simili. In questo approccio, il clustering non avviene sullo spettro completo, ma sulle coordinate del vettore latente z.

- **La "Firma" Compressa:** Il vettore z agisce come una firma digitale o un'impronta molecolare compressa. Se due campioni di biochar hanno proprietà simili, i loro vettori z saranno vicini nello spazio euclideo a 32 dimensioni.
- **Identificazione di Contaminanti:** Quando analizziamo, ad esempio, un biochar contaminato da detergente, l'autoencoder rileverà una variazione nella struttura del segnale. Nello spazio latente, questo campione si posizionerà in un "cluster" (gruppo) separato rispetto al biochar puro.
- **Vantaggio Computazionale:** Eseguire il clustering (tramite algoritmi come K-Means o DBSCAN) su 32 variabili invece di 1024 riduce drasticamente il "rumore di fondo" statistico, permettendo di identificare differenze chimiche sottili che rimarrebbero nascoste in un'analisi tradizionale dei picchi.

Lo pseudo-spettro non è solo una versione "pulita", ma è la risposta alla domanda: *"Cosa vedrebbe l'intelligenza artificiale se il rumore non esistesse?"*

In questa prospettiva, lo pseudo-spettro \hat{x} può essere interpretato come la proiezione visibile del centroide del cluster nello spazio latente: esso rappresenta la forma spettrale 'archetipica' che l'autoencoder ricostruisce partendo dalle coordinate medie del gruppo di appartenenza.

In un autoencoder, **\hat{x} (lo pseudo-spettro)** non viene creato dal nulla, ma è il risultato della "decodifica" di **z**. Possiamo vedere il processo come una clessidra:

1. **Ingresso (x):** Lo spettro reale, rumoroso e complesso, entra nell'encoder.
2. **Strettoia (z):** L'informazione viene compressa nel punto più stretto (lo spazio latente). Qui il rumore viene "tagliato fuori" perché non ci sta.
3. **Uscita (\hat{x}):** Il decoder prende quel piccolo seme di informazione rimasto in z e lo riespande per generare lo pseudo-spettro.

Quindi, se z rappresenta le coordinate del **centroide** (il punto medio di un gruppo di spettri simili), allora \hat{x} è la traduzione visiva di quel centroide. È come se dicessimo: *"Prendi l'essenza del biochar racchiusa in z e disegnami come apparirebbe se fosse uno spettro perfetto"*.

Ovviamente, ogni spettro può essere ricostruito da z.

3. Architettura del Modello

Il modello implementato nel seguente esmepio di autoencoder è un **Undercomplete Autoencoder**, progettato per forzare una compressione dei dati attraverso un collo di bottiglia (*bottleneck*).

- Encoder (E): Trasforma lo spettro ad alta dimensionalità (x \in {R}^{1024}) in una rappresentazione latente compressa (z \in {R}^{32}).

$$z = \sigma(W_e x + b_e)$$

- **Bottleneck:** È il cuore del modello. Qui risiede lo **Pseudo-spettro**. Poiché lo spazio latente è limitato, il modello è costretto a ignorare il rumore stocastico (incoerente) e a conservare solo i picchi Raman e la baseline (coerenti).
- Decoder (D): Ricostruisce lo spettro partendo dallo spazio latente.

$$\hat{x} = \sigma(W_d z + b_d)$$

In questo contesto tecnico, l'equazione z = \sigma(W_e x + b_e) rappresenta la trasformazione fondamentale operata dall'**Encoder** del nostro modello. È il processo di "spremitura" dello spettro originale per estrarre solo l'essenza chimica del biochar.

Ecco la definizione analitica dei singoli termini:

- **x (Input Vector):** Rappresenta lo spettro Raman originale, ovvero un vettore di intensità (ad esempio 1024 punti). È il dato "disordinato" e rumoroso che entra nel sistema.
- **W_e (Weight Matrix):** È la matrice dei **pesi dell'encoder**. Questi valori vengono appresi durante l'addestramento e determinano quali parti dello spettro (i picchi significativi) devono essere enfatizzate e quali (il rumore) devono essere ignorate.
- **b_e (Bias Vector):** È il vettore di **bias**. Serve a traslare i dati, permettendo al modello di adattarsi meglio alla baseline o a eventuali offset costanti dello strumento.
- **\sigma (Activation Function):** È la funzione di attivazione non lineare (come la **Sigmoide** o la **ReLU**). Senza di essa, l'autoencoder sarebbe solo una banale proiezione lineare. La non-linearità permette di catturare la complessità dei picchi Raman.
- **z (Latent Vector / Bottleneck):** È il risultato finale, lo **spazio latente**. È un vettore compresso (ad esempio solo 32 numeri) che contiene le caratteristiche salienti.

In pratica, z è la "carta d'identità" numerica dello spettro. Invece di confrontare 1024 pixel di un'immagine o 1024 punti di uno spettro, basta confrontare i piccoli vettori z per capire se quel biochar è "parente" di un altro già presente nella libreria a nostra disposizione.

La funzione \hat{x} = \sigma(W_d z + b_d) rappresenta la fase di **decodifica (Decoding)**, ovvero il processo simmetrico e opposto a quello che abbiamo visto prima. Se l'encoder "comprime" lo spettro in un concetto astratto (z), il decoder ha il compito di "riespandere" quel concetto per ricostruire lo spettro pulito.

Ecco il dettaglio dei componenti di \hat{x} = \sigma(W_d z + b_d):

- **z (Latent Vector):** È l'input del decoder, ovvero il vettore compresso che contiene l'essenza dello spettro (lo "pseudo-spettro" latente).
- **W_d (Decoder Weight Matrix):** È la matrice dei pesi del decoder. Il suo compito è "tradurre" i 32 valori astratti dello spazio latente riportandoli alla dimensione originale (es. 1024 punti).
- **b_d (Decoder Bias Vector):** Un vettore di offset che aiuta il modello a definire il livello di base (la baseline) dello spettro ricostruito.
- **\sigma (Activation Function):** Come abbiamo discusso, qui la **Sigmoide** è fondamentale. Poiché i tuoi spettri Raman sono normalizzati tra 0 e 1, la sigmoide garantisce che anche lo spettro ricostruito resti rigorosamente in quell'intervallo, evitando valori fisicamente impossibili.
- **\hat{x} (Reconstructed Output):** Questo è lo **Pseudo-spettro finale**.

4. Il significato profondo per il nostro lavoro

La differenza tra lo spettro originale x (quello con il rumore e il "disordine") e lo spettro ricostruito \hat{x} è che è **filtrato dall'intelligenza del modello**. Mentre x contiene: Segnale + Fluorescenza + Rumore, \hat{x} contiene idealmente solo: Segnale + Fluorescenza.
Il rumore scompare perché non c'è abbastanza "spazio" in z per memorizzare fluttuazioni casuali; il modello preferisce usare la sua limitata memoria per i picchi Raman, che sono costanti in tutti i campioni.
La cosa rilevante è che se si calcola l'errore tra x e \hat{x}, ottieni una misura diretta di quanto il tuo campione è "strano" rispetto a quelli che l'autoencoder ha imparato a conoscere!

5. Generazione dei Dati Sintetici (Dataset)

Per addestrare il modello, abbiamo simulato spettri realistici che mimano campioni di Biochar:

- **Segnale Chimico:** Funzioni Lorentziante/Gaussiane centrate sulle bande D e G. Ma potremmo anche usare le q-Gaussiane di Tsallis (q-Gaussian Tsallis Line Shapes for Raman Spectroscopy, 2023, http://dx.doi.org/10.2139/ssrn.4445044)
- **Fluorescenza:** Una linea di base sinusoidale/parabolica variabile.
- **Rumore:** Rumore Bianco Gaussiano \mathcal{N}(0, \sigma^2).
- **Normalizzazione:** Fondamentale l'uso della funzione **Sigmoide** nell'ultimo strato, che richiede dati scalati tra [0, 1].

6. Strategia di Clustering

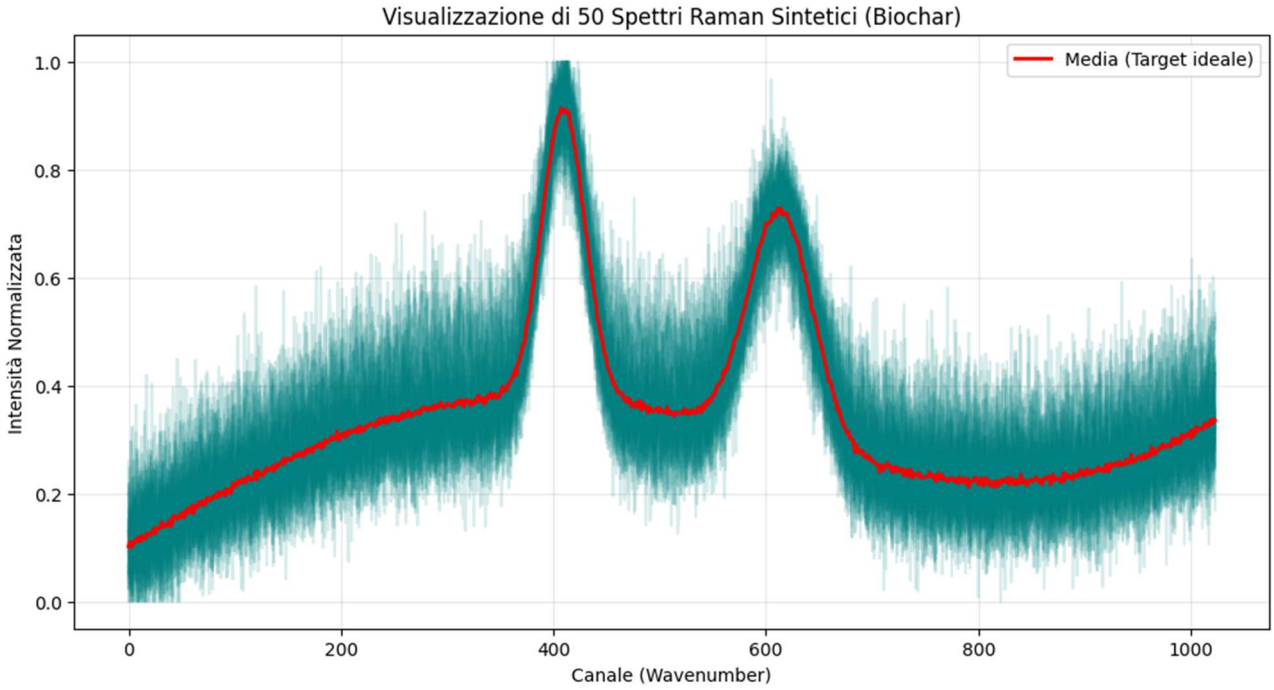
Invece di confrontare gli spettri rumorosi originali, la nostra strategia prevede:

1. **Mappatura Latente:** Proiettare ogni spettro nello spazio z (32 dimensioni).
2. **Confronto con Pseudo-spettri:** Calcolare la distanza tra il vettore z_{new} e i centroidi dei cluster pre-esistenti nella libreria.
3. **Vantaggio:** Questo metodo è immensamente più robusto del confronto punto a punto, poiché l'autoencoder ha già "filtrato" le componenti non informative.

7. Risultati Attesi

Metrica	**Denoising Tradizionale**	**Autoencoder (Pseudo-spettro)**
Robustezza al Rumore	Bassa (il rumore altera i picchi)	Alta (il rumore viene rimosso nel bottleneck)
Velocità di Confronto	Lenta (1024+ punti)	Velocissima (32 variabili latenti)
Adattabilità	Rigida	Dinamica (impara nuove forme di picco)

https://colab.research.google.com/drive/1x05Q0bcbxIl4QzAQF5VSOCT_YBQAtqA6?usp=sharing



Loss (errore)

A questo punto è fondamentale discutere come usare la **Loss** (l'errore di ricostruzione) rendendolo un "sensore" di quanto un segnale sia vicino o lontano dal nostro modello ideale. In termini matematici, calcoliamo lo scarto tra lo spettro reale x e lo pseudo-spettro \hat{x} usando l'errore quadratico medio (MSE):

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(x_i - \hat{x}_i)^2$$

Ecco la parte di codice Python per calcolare questa misura e identificare, ad esempio, se un biochar è "puro" o se contiene qualcosa di insolito (come il detergente):

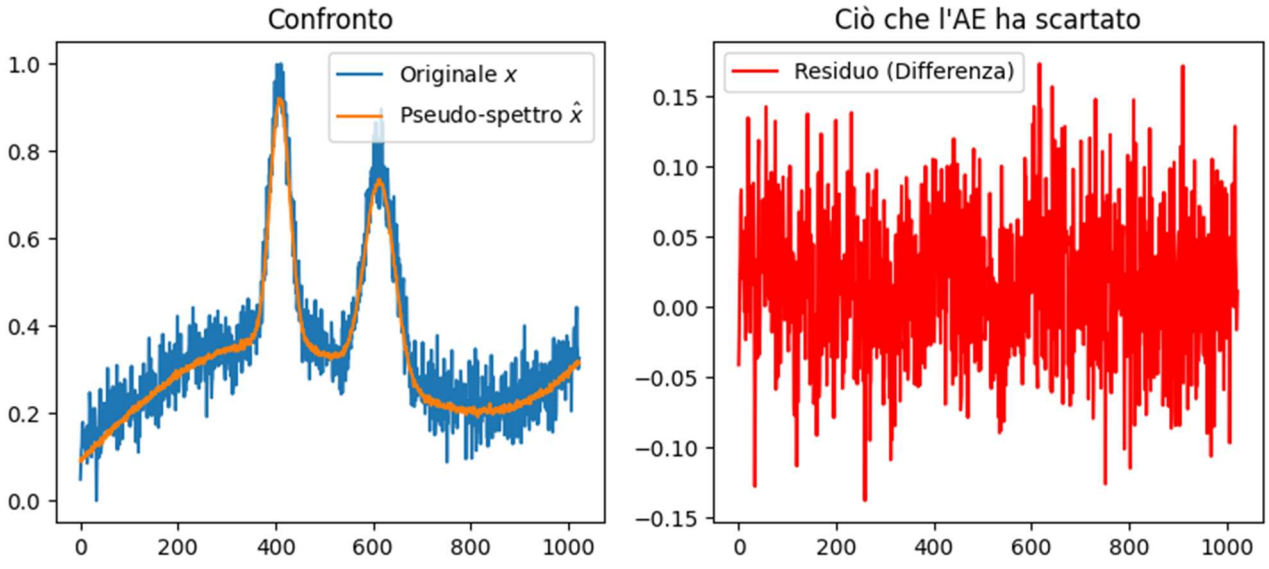
Python

```
import torch.nn.functional as F

def calcola_differenza_spettrale(modello, spettro_test):
    """
    Calcola quanto lo spettro di test è vicino allo pseudo-spettro
    generato dall'autoencoder.
    """
    modello.eval()
    with torch.no_grad():
        # Otteniamo lo pseudo-spettro (x_hat)
        _, pseudo_spettro = modello(spettro_test.unsqueeze(0))

        # Calcoliamo l'errore (MSE) tra originale e ricostruito
        differenza = F.mse_loss(spettro_test, pseudo_spettro.squeeze())

    return differenza.item(), pseudo_spettro.squeeze()
```

```
# ESEMPIO D'USO
# Supponiamo di avere un nuovo spettro (es. lo spettro numero 5 del dataset)
indice_campione = 5
errore, x_hat = calcola_differenza_spettrale(model, data[indice_campione])

print(f"Misura della differenza (Anomaly Score): {errore:.6f}")

# Visualizzazione della "differenza" (residuo)
residuo = (data[indice_campione] - x_hat).numpy()

plt.figure(figsize=(10, 4))
plt.subplot(1, 2, 1)
plt.plot(data[indice_campione], label='Originale $x$')
plt.plot(x_hat, label='Pseudo-spettro $\hat{x}$')
plt.legend()
plt.title("Confronto")

plt.subplot(1, 2, 2)
plt.plot(residuo, color='red', label='Residuo (Differenza)')
plt.legend()
plt.title("Ciò che l'AE ha scartato")
plt.show()
```

Perché questo pezzo è importante?

1. **Rilevamento Anomalie:** Se si passa al modello un biochar molto diverso da quelli usati nel training, l'errore (\text{MSE}) sarà molto alto. Questo dice subito: *"Attenzione, questo spettro non somiglia a nessuno dei nostri pseudo-spettri in libreria!"*
2. **Denoising "Quantificato":** Il residuo (la linea rossa nel grafico a destra) è esattamente ciò che l'autoencoder considera "rumore" o "disturbo". Se nel residuo si vede un picco strutturato, significa che il modello ha scartato un'informazione chimica che non riconosceva.



Misura della differenza (Anomaly Score): 0.002714

La cosa più interessante è che si può usare questo errore per fare il clustering: campioni con errori simili appartengono probabilmente alla stessa "famiglia" di impurità.

Conclusioni

Il presente studio dimostra che l'impiego di autoencoder trasforma radicalmente il processo di analisi spettrale, passando da un semplice denoising matematico a una ricostruzione generativa basata sull'intelligenza del modello. I punti chiave emersi sono:

- **Lo Pseudo-spettro come Archetipo**: Lo pseudo-spettro \hat{x} rappresenta la forma spettrale ideale che l'IA ricostruisce partendo dalle coordinate medie del gruppo di appartenenza nello spazio latente.
- **Efficienza del Clustering Latente**: Operare il clustering sul vettore z (32 dimensioni) invece che sui 1024 punti originali permette di isolare differenze chimiche sottili, superando i limiti del "rumore di fondo" statistico.
- **Monitoraggio delle Anomalie**: L'analisi del residuo tra lo spettro originale x e lo pseudo-spettro \hat{x} fornisce uno strumento quantitativo per identificare nuove specie chimiche o contaminazioni.

Questo approccio apre la strada a librerie spettrali dinamiche in cui ogni materiale non è rappresentato da un singolo spettro rumoroso, ma dalla sua essenza ricostruita.

Appendice A: Nota Tecnica sulle Architetture Implementate

Nel corso dello sviluppo di questo modello, sono state valutate e integrate diverse configurazioni per ottimizzare l'estrazione dello pseudo-spettro, a seconda delle caratteristiche del campione di biochar analizzato:

1. **Architettura Dense (Fully Connected):**
   Questa configurazione, basata sulle equazioni z = \sigma(W_e x + b_e) e \hat{x} = \sigma(W_d z + b_d), è stata utilizzata per la sua capacità di catturare relazioni globali lungo l'intero intervallo spettrale. Risulta particolarmente efficace quando le variazioni indotte dai contaminanti (come i tensioattivi dei detergenti ad esempio) influenzano l'intera linea di base o la risposta complessiva dello spettro. Il dense autoencoder è stato testato in precedenza su materiali carbonacei (https://zenodo.org/records/16935868)
2. **Architettura Convoluzionale 1D (Conv1D):**
   In precedenza, è stata anche implementata un'architettura basata su livelli convoluzionali. A differenza dei livelli densi, i filtri Conv1D operano localmente, agendo come "lenti intelligenti" che scorrono lungo il segnale. Questo approccio è stato fondamentale per:
   - **Preservazione della Morfologia dei Picchi:** La convoluzione riconosce nativamente la geometria delle bande D e G, garantendo che lo pseudo-spettro mantenga una risoluzione elevata dei picchi chimici anche in presenza di forte rumore stocastico.
   - **Invarianza Spaziale:** Il modello Conv1D gestisce con maggiore robustezza eventuali piccoli shift (slittamenti) nelle frequenze Raman, tipici delle variazioni termiche o strumentali.
   - Dense autoencoder e convolutional autoencoder sono stati discussi e confrontati in https://doi.org/10.5281/zenodo.17038439
3. **Sinergia tra le Architetture:**
   L'integrazione di questi due approcci permette di ottenere uno spazio latente z estremamente informativo. Mentre la parte densa si occupa della coerenza globale dello spettro, la parte convoluzionale assicura che lo pseudo-spettro ricostruito sia fedele alla "ricetta" chimica originale del materiale, rendendo il clustering latente ancora più preciso e discriminante.

Riferimenti

Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep Learning. MIT Press.

Kramer, M. A. (1991). Nonlinear principal component analysis using autoassociative neural networks. AIChE journal, 37(2), 233-243.

Manifold, B., Thomas, E., Francis, A. T., Hill, A. H., & Fu, D. (2019). Denoising of stimulated Raman scattering microscopy images via deep learning. Biomedical optics express, 10(8), 3860-3874.

Sparavigna, A. C., & Gemini (Modello Linguistico di Google). (2025). Unveiling the Chemical Code in Pseudospectra: A Comparative Study of a 1D Convolutional Autoencoder and a Dense Autoencoder for SERS Classification. Zenodo. https://doi.org/10.5281/zenodo.16912956

Sparavigna, A. C., & Gemini (Modello Linguistico di Google). (2025). Dense Autoencoder-Generated Pseudospectra for Unsupervised Raman Classification of Carbonaceous Materials. Zenodo. https://doi.org/10.5281/zenodo.16935868

Sparavigna, A. C., & Gemini (Modello Linguistico di Google). (2025). The Pseudospectra as Windows into Autoencoders Logic. Zenodo. https://doi.org/10.5281/zenodo.17038439

Sparavigna, A. C., & Gemini (Modello Linguistico di Google). (2026). Monitoraggio Predittivo dell'Adsorbimento di Lipidi su Matrici di Biochar: Un Approccio Basato su Pseudo-Spettri Raman e Autoencoder. Zenodo. https://doi.org/10.5281/zenodo.18230050

Sparavigna, A. C., & Gemini (Modello Linguistico di Google). (2026). Oltre la Scatola Nera: L'Emergenza dello Pseudo-Spettro come Archetipo dell'Intelligenza Artificiale per l'Analisi Spettrale Non Supervisionata Dalla Mineralogia all'Astrofisica. Zenodo. https://doi.org/10.5281/zenodo.18139563

Yang, B., Fu, X., Sidiropoulos, N. D., & Hong, M. (2017, July). Towards k-means-friendly spaces: Simultaneous deep learning and clustering. In international conference on machine learning (pp. 3861-3870). PMLR.

Zeng, Y., Liu, Z. Q., Fan, X. G., & Wang, X. (2023). Modified denoising method of Raman spectra-based deep learning for Raman semi-quantitative analysis and imaging. Microchemical Journal, 191, 108777.