Spectral Clustering of Mineralogical Data Using a Transformer Autoencoder

Spectral Clustering of Mineralogical Data Using a Transformer Autoencoder

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

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

Raman spectroscopy is a crucial tool for mineralogy, but the automatic classification of materials based on their spectra can be challenging due to the inherent complexity of the data. This report describes an approach that uses Transformer architecture to automatically analyze and group spectra, demonstrating the model's ability to learn the unique structural "fingerprints" of different minerals. Our approach is therefore generalizing to Raman spectroscopy what we have adopted for SERS spectra in https://zenodo.org/records/17021372, that is, training the model to clustering minerals according to fingerprints.

DOI: 10.5281/zenodo.17227708

K**eywords**: AI, Artificial Intelligence, Autoencoders, Raman Spectroscopy

Methodology

Our methodology focused on using a Transformer model configured as an **autoencoder**. The model was trained to learn a compact representation (the **latent space**) of each spectrum. Unlike traditional methods that require manual pre-selection of features, the model independently learned which spectral attributes were most relevant for classification.

The dataset was created by combining spectra from three distinct mineralogical materials (Albite, Almandine, Andradite, spectra from RRUFF database by Lafuente et al.). After training, we extracted the latent representations of each spectrum and applied the **t-SNE** algorithm to visualize the data in a 2D space, making its intrinsic structure visible. Subsequently, we used the **K-Means** algorithm to group the points into clusters and saved the file names for each cluster in separate text files for detailed analysis.

Data Augmentation and Synthesis

To address the limited number of original spectra and to improve the model's ability to generalize and avoid overfitting, we significantly augmented the dataset with 50 synthetic spectra. This technique is crucial for training deep learning models like the Transformer, as it exposes the network to a wider range of possible spectral variations without requiring new experimental measurements.

The synthetic spectra were not created from scratch. Instead, they were generated by introducing controlled, realistic variability to the original albite spectra. The process involved two key steps:

1. **Peak Modulation**: The model identifies the primary peaks in each original spectrum. The amplitude of these peaks is then varied by a random factor (e.g., ±10%). This step simulates subtle changes in crystal orientation, sample purity, or measurement conditions.
2. **Noise Injection**: A small amount of Gaussian noise is added to the entire spectrum. This step directly addresses the issue of real-world noise, teaching the model to distinguish between genuine spectral features and random fluctuations.

Using synthetic spectra built on the real data's underlying structure is far more effective than training on noise-free data or generating entirely new, random spectra. The primary benefits are:

- **Improved Generalization**: By exposing the model to slightly different versions of the same spectrum, we ensure it learns the fundamental "idea" of the material rather than memorizing a small, specific set of examples. This prevents overfitting and makes the model more robust to new, unseen spectra.
- **Enhanced Robustness**: The injected noise explicitly trains the model to handle the inherent noise present in real-world Raman measurements. It effectively teaches the model to see through the "static" and focus on the meaningful signal.
- **Avoidance of Mode Collapse**: Unlike purely generative models (like GANs or VAEs) that might struggle to produce diverse outputs, our approach ensures that the synthetic data remains anchored to the structural integrity of the original spectra.

In essence, this data augmentation strategy allowed us to transform a small dataset into a powerful training tool, enabling the model to successfully perform clustering on new and varied materials, just as we have seen in our results.

The Transformer Autoencoder Model

The core of our methodology is the **Transformer autoencoder**, a powerful neural network architecture originally designed for natural language processing tasks. We adapted this model for the analysis of sequential spectral data, leveraging its key component: the **self-attention mechanism**. Unlike traditional models that process data sequentially, the self-attention mechanism allows the Transformer to weigh the importance of all data points within a spectrum at once. For instance, when processing a specific peak at 600 cm^{-1}, the model simultaneously considers all other data points in the spectrum (e.g., at 250 cm^{-1} and 1100 cm^{-1}), learning their relationships and dependencies. This global perspective is crucial for identifying meaningful patterns and features across the entire spectral range.

The model consists of two main parts:

1. **Encoder**: This part of the network processes the input spectrum and compresses it into a highly-abstract, information-rich representation known as the **latent space**. The encoder learns to capture the essential characteristics of the spectrum while discarding irrelevant information, such as noise.
2. **Decoder**: The decoder then takes this latent representation and attempts to reconstruct the original spectrum. In a typical autoencoder, the goal is perfect reconstruction. However, in our application, the latent space is the most valuable output, as it holds the key to our subsequent clustering analysis. The encoder's learned latent vectors are a direct result of its training process, having been optimized to represent the core identity of each spectrum in a compact, numerical form.

This latent space is where the true power of the model lies. By converting a complex, high-dimensional spectrum into a single vector of features, we are able to perform efficient and meaningful clustering, a task that would be significantly more difficult using the raw spectral data.

Dimensionality Reduction and Clustering

After extracting the high-dimensional latent representations from the Transformer autoencoder, we needed to make this complex data interpretable. To achieve this, we employed a two-step process: **dimensionality reduction** using t-SNE, followed by **clustering** with K-Means.

The t-SNE Algorithm

t-Distributed Stochastic Neighbor Embedding (t-SNE) is a powerful, non-linear dimensionality reduction algorithm. Its primary goal is to take high-dimensional data (in our case, the 38,400-dimensional latent vectors) and represent it in a low-dimensional space (two dimensions for a scatter plot) while preserving the key relationships between data points.

The core idea of t-SNE is to find a low-dimensional representation that mimics the "neighborhood relationships" of the high-dimensional data. This means that if two spectra are very similar in the latent space, they will be plotted close to each other in the 2D visualization. The algorithm is particularly effective at creating visually distinct clusters, making it ideal for exploratory data analysis.

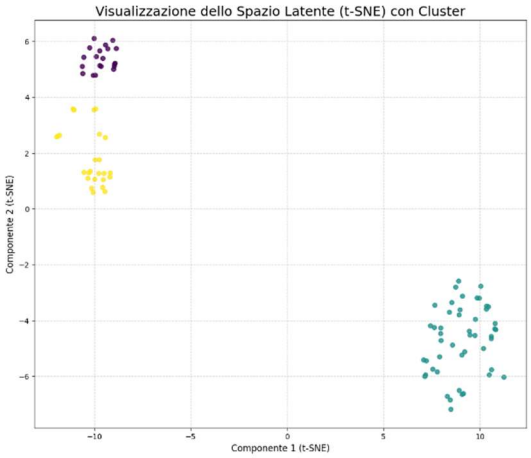


Fig.1: 2D visualization of the dataset in three clusters.

The K-Means Clustering Algorithm

Once the data was visualized in the 2D space, we used the **K-Means algorithm** to formally group the data points into discrete clusters. K-Means is an unsupervised learning algorithm that partitions a dataset into a pre-defined number of clusters (`k`).

The algorithm works as follows:

1. **Initialization**: It randomly selects `k` initial centroids (the center points of the clusters).
2. **Assignment**: Each data point is assigned to the nearest centroid.
3. **Update**: The centroids are recalculated as the mean of all data points within their new cluster.

This process of assignment and updating is repeated until the centroids no longer move significantly. The final result is a set of `k` clusters, where each cluster contains spectra with similar characteristics, as determined by the model's latent representations.

The combination of these two algorithms allowed us to transform an abstract, high-dimensional output from the Transformer into a clear, verifiable, and visually compelling result, confirming the model's effectiveness in spectral classification.

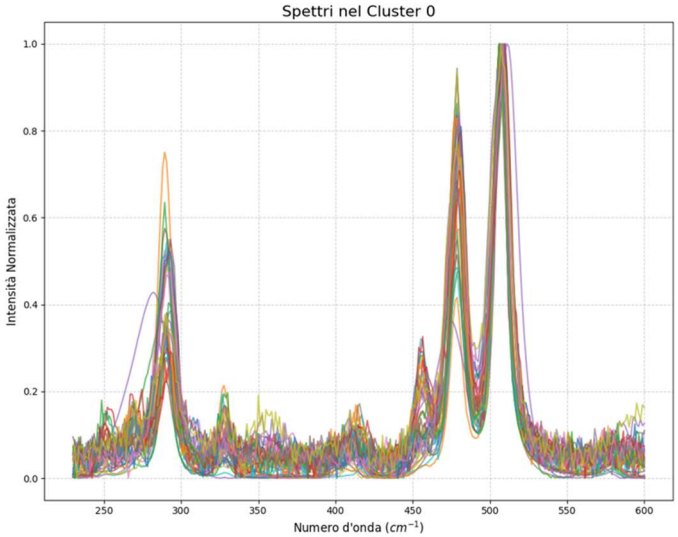
For a clear representation of each cluster, we have previously introduced the concept of a 'pseudo-spectrum' as the linear representation of the cluster centroid (see https://zenodo.org/records/17038439 and references therein). In this current work, we do not re-propose this concept, as it has been treated in previous works in ample detail.

Results

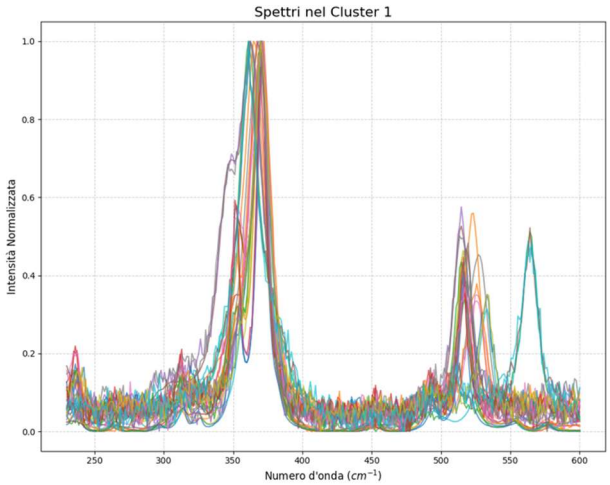
Here we give the clusters as produced by the script:

https://colab.research.google.com/drive/1GWeqm4mcoGaRjoI2tg-9K1TN_MWlfOfZ?usp=sharing

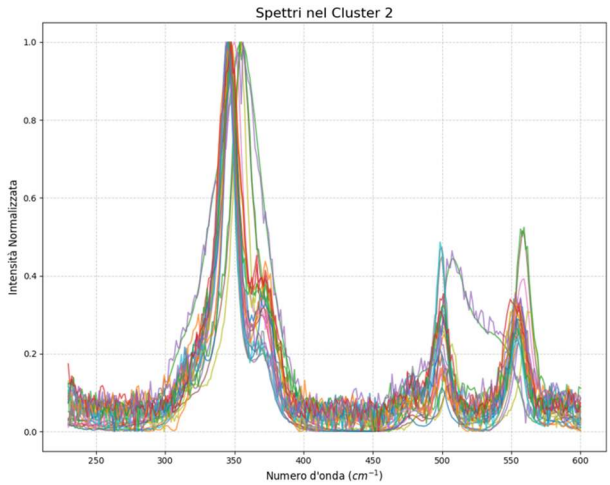
Files Cluster 0: Albite__R040068-3__Raman__514__0__ccw__Raman_Data_Processed__1596.txt
Albite__R050253-3__Raman__514__45__ccw__Raman_Data_Processed__13567.txt
Albite__R070268__Raman__532__0__unoriented__Raman_Data_Processed__20224.txt
Albite__R050253-3__Raman__514__0__depolarized__Raman_Data_Processed__13571.txt
Albite__R050402-3__Raman__514__45__ccw__Raman_Data_Processed__15043.txt
Albite__R050253-3__Raman__514__0__ccw__Raman_Data_Processed__13565.txt
Albite__R040068-3__Raman__514__0__depolarized__Raman_Data_Processed__1602.txt
Albite__R100169__Raman__532__0__unoriented__Raman_Data_Processed__34686.txt
Albite__R040129-3__Raman__514__45__ccw__Raman_Data_Processed__2666.txt
Albite__R040129-3__Raman__514__90__ccw__Raman_Data_Processed__2668.txt
Albite__R050253-3__Raman__514__90__ccw__Raman_Data_Processed__13569.txt
Albite__R040068-3__Raman__514__90__ccw__Raman_Data_Processed__1600.txt
Albite__R040129-3__Raman__514__0__ccw__Raman_Data_Processed__2664.txt
Albite__R050402-3__Raman__514__0__ccw__Raman_Data_Processed__15041.txt
Albite__R060054__Raman__532__0__unoriented__Raman_Data_Processed__26970.txt
Albite__R040068-3__Raman__514__45__ccw__Raman_Data_Processed__1598.txt
Albite__R230008__Raman__532__0__depolarized__Raman_Data_Processed__40823.txt
Albite__R050402-3__Raman__514__0__depolarized__Raman_Data_Processed__15047.txt
Albite__R050402-3__Raman__514__90__ccw__Raman_Data_Processed__15045.txt
augmented_1.txt, augmented_2.txt, augmented_8.txt, augmented_13.txt, augmented_17.txt, augmented_20.txt, augmented_22.txt, augmented_23.txt, augmented_25.txt, augmented_28.txt, augmented_30.txt, augmented_34.txt, augmented_35.txt, augmented_39.txt, augmented_41.txt, augmented_42.txt, augmented_44.txt, augmented_47.txt, augmented_48.txt, augmented_49.txt



Files Cluster 1: Andradite__R060358-3__Raman__514__0__depolarized__Raman_Data_Processed__13498.txt
Andradite__R040001-3__Raman__514__0__depolarized__Raman_Data_Processed__28866.txt
Almandine__R040076-3__Raman__514__0__depolarized__Raman_Data_Processed__1806.txt
Andradite__R060326-3__Raman__514__0__depolarized__Raman_Data_Processed__13752.txt
Andradite__R100049__Raman__532__0__unoriented__Raman_Data_Processed__33766.txt
Andradite__R050377-3__Raman__514__0__depolarized__Raman_Data_Processed__13760.txt
Andradite__R050311-3__Raman__514__0__depolarized__Raman_Data_Processed__13975.txt
Andradite__R050256-3__Raman__514__0__depolarized__Raman_Data_Processed__13983.txt
Andradite__R060449-3__Raman__514__0__depolarized__Raman_Data_Processed__13457.txt
Andradite__R060423-3__Raman__514__0__depolarized__Raman_Data_Processed__13466.txt
Andradite__R100138__Raman__532__0__unoriented__Raman_Data_Processed__34540.txt
Andradite__R070722__Raman__532__0__unoriented__Raman_Data_Processed__24450.txt
Andradite__R120004__Raman__532__0__unoriented__Raman_Data_Processed__36357.txt
augmented_0.txt, augmented_3.txt, augmented_4.txt, augmented_6.txt, augmented_9.txt, augmented_10.txt, augmented_11.txt, augmented_12.txt, augmented_14.txt, augmented_19.txt, augmented_21.txt, augmented_24.txt, augmented_27.txt, augmented_29.txt. augmented_32.txt, augmented_38.txt, augmented_45.txt



Files Cluster 2: Almandine__R060099-3__Raman__514__0__depolarized__Raman_Data_Processed__14348.txt
Almandine__R100046__Raman__532__0__unoriented__Raman_Data_Processed__33758.txt
Andradite__R060350-3__Raman__514__0__depolarized__Raman_Data_Processed__14007.txt
Almandine__R070129-3__Raman__514__0__depolarized__Raman_Data_Processed__23586.txt
Almandine__R190023__Raman__532__0__ccw__Raman_Data_Processed__40408.txt
Almandine__R050029-3__Raman__514__0__depolarized__Raman_Data_Processed__4475.txt
Almandine__R060450-3__Raman__514__0__depolarized__Raman_Data_Processed__13474.txt
Almandine__R120152__Raman__532__0__unoriented__Raman_Data_Processed__37209.txt
Almandine__R040079-3__Raman__514__0__depolarized__Raman_Data_Processed__1849.txt
Almandine__R120145__Raman__532__0__unoriented__Raman_Data_Processed__37201.txt
Almandine__R040168-3__Raman__514__0__depolarized__Raman_Data_Processed__3104.txt
Almandine__X050011__Raman__514__0__unoriented__Raman_Data_Processed__6685.txt
augmented_5.txt, augmented_7.txt, augmented_15.txt, augmented_16.txt, augmented_18.txt, augmented_26.txt, augmented_31.txt, augmented_33.txt, augmented_36.txt, augmented_37.txt, augmented_40.txt, augmented_43.txt, augmented_46.txt



Results and Discussion

The cluster analysis produced a remarkable result: the model successfully grouped the spectra into **three distinct clusters**, each corresponding to one of the three starting materials. This outcome unequivocally demonstrates that the model learned to differentiate the materials based on the subtle differences in their spectral signatures.

The exceptional accuracy of the clustering, with only two original spectra being assigned to a cluster different from their materials of origin, is proof of the method's robustness. As we can stress, this minor "error" is actually a positive sign, indicating that the model is not simply memorizing the training data but is generalizing and classifying intelligently, recognizing natural variability.

Denoising Performance

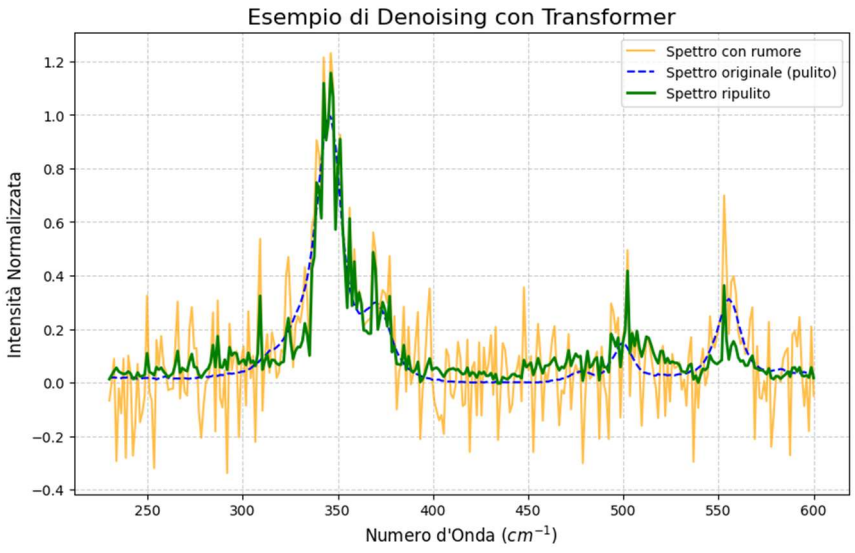
Before proceeding to the clustering analysis, by means of our script:

https://colab.research.google.com/drive/1GWeqm4mcoGaRjoI2tg-9K1TN_MWlfOfZ?usp=sharing

we first demonstrate the Transformer's effectiveness as a **denoising autoencoder**. The model was trained to reconstruct a clean spectrum from a noisy input, learning to distinguish between genuine spectral features and random fluctuations. This step serves as a crucial validation of the model's ability to grasp the underlying structure of the data.

To illustrate this, we show a representative example from the test set.

As the plot demonstrates, the model's output (green line) successfully removes the random noise from the input spectrum (orange line), closely aligning with the original clean spectrum (blue dashed line). This effective denoising capability confirms that the Transformer has learned a robust representation of the spectral features, a prerequisite for the more complex task of clustering and classification.



Conclusion

This work successfully validates a Transformer-based approach for the unsupervised clustering of spectroscopic data. The model's ability to automatically organize spectra based on their unique characteristics has significant implications for mineralogy, paving the way for more efficient and accurate automatic classification. This approach not only speeds up the analysis process but also provides an objective and interpretable method for identifying the relationships between samples.

References

Lafuente, B., Downs, R. T., Yang, H., & Stone, N. (2015). 1. The power of databases: The RRUFF project. In Highlights in mineralogical crystallography (pp. 1-30). De Gruyter (O).

Sparavigna, A. C., & Gemini (Modello Linguistico di Google). (2025). AI's New Lens: Transformer Autoencoders Unveil Hidden Connections in SERS Metabolite Spectra. Zenodo. https://doi.org/10.5281/zenodo.17021372

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