Unsupervised Learning - final assignment

Unsupervised Image Clustering comparison with PCA, NMF, Autoencoder and DEC

by M. Giordano

Introduction

This is an Unsupervised Learning task focused on Image Clustering comparison with different unsupervised models.
The dataset is composed of various photorealistic natural AI generated images .
Our goal is to build a series of models to compare their ability to correctly clusterize into three groups: animals, birds and landscapes.

This will be achieved through EDA, data cleaning, feature engineering, and model development from scratch.

This work covers all the required points of the rubric, following the steps in the same order of appearance:

1. Gather data, determine the method of data collection and provenance of the data (3 points)

2. Identify an Unsupervised Learning Problem (6 points)

3. Exploratory Data Analysis (EDA) — Inspect, Visualize, and Clean the Data (26 points)

4. Perform Analysis Using Unsupervised Learning Models, Present Discussion, and Conclusions (70 points)

5. Produce Deliverables: High-Quality, Organized Jupyter Notebook Report, Video Presentation, and GitHub Repository (35 points)

You can find this document, the video presentation and the jupiter notebook.ipynb at the following Github repository:

https://github.com/michele-giordano/unsupervised_learning_final_assignment/

Step 1: Gather data, determine the method of data collection and provenance of the data

1.1 Data Collection and Provenance

The dataset used in this work was created directly by the author, taking initial inspiration from image clustering competitions found on Kaggle.
During the review phase, I noticed that the datasets available there were generally of poor visual quality and highly inconsistent, often containing noisy, low-resolution, or poorly framed images. These datasets reached a maximum accuracy values around 0.8 even under supervised learning conditions, indicating that the visual variability present was not informative but mostly chaotic.

For this reason, they were not considered suitable for an unsupervised learning task where the structure of the data itself must guide the formation of meaningful clusters.

To address this limitation, Stable Diffusion was used to generate a new dataset with a similar photographic style, but with controlled visual quality.
The images retain natural and realistic appearance, and include subjects that are not trivially separable, yet are clearly represented. This approach ensures a dataset that is coherent, reproducible, and appropriate for evaluating unsupervised image clustering methods.

1.2 Method of Data Collection

The images were generated with the explicit goal of representing the subjects in a structured and classifiable manner, while maintaining a high degree of heterogeneity within the dataset. More than five hundred images were created from scratch, including a variety of animal species, different types of birds, and diverse landscape scenarios across multiple seasons and lighting conditions. The intent was to build a dataset that is visually coherent but not uniform, allowing for meaningful variation along relevant visual features.

Considering that the unsupervised methods used in this work will attempt to form clusters directly from pixel-level information, several images were deliberately designed to be challenging from a clustering perspective. Examples include polar bears against snowy backgrounds or brown-fur animals placed in forest environments, , where subject and background share very similar color distributions and silhouette is less distinct.

These controlled difficulties were introduced to prevent the dataset from being trivially separable and to ensure that the clustering methods must effectively extract and leverage visual structure rather than relying on simple color or texture cues.

Step 2: Identify the Unsupervised Learning Problem

This is an unsupervised task of image clustering based on photorealistic images.
The goal is to conduct EDA, perform data cleaning, extract meaningful visual features, and train a clustering model able to assign images to one of three conceptual groups:

- animals,
- birds,
- landscapes.

The model does not receive labeled data during training; instead, the structure must emerge from the visual patterns present in the images themselves. The evaluation therefore focuses on whether the resulting clusters form coherent visual groupings that align with these three categories, without relying on supervision or ground-truth labels.

This work goes through all the rubric requirements, implementing the necessary code entirely from scratch.
The analysis is designed to independently explore different approaches to feature extraction, dimensionality reduction, and clustering, rather than reproducing any existing kernels that could be found on Kaggle on similar tasks.

Each stage of the pipeline, from preprocessing to model comparison, is developed and validated within this notebook to ensure methodological originality and full adherence to the rubric's expectations for analytical depth and reproducibility.

Step 3: Exploratory Data Analysis (EDA) - Inspect, Visualize, and Clean the Data

3.1 Describe the factors or components that make up the dataset

As a first operation, we inspect the size and shape of the dataset to understand the nature of the collected data.
This step is essential to verify data integrity, identify potential irregularities, and gain an initial intuition about the structure of the images.

```
Total number of images: 577
Minimum dimension: 44 x 108
Maximum dimension: 256 x 256
Number of distinct resolutions: 3
```

We can see that the dataset consists of 577 images with resolutions that range from 44 x 108 to 256 x 256.

This diversity is intentional, as it focuses the attention on doing a proper data cleaning. Since the dataset is declared having 256x256, all other dimensions are considered noise to be removed.

Each image can be represented as a three-dimensional matrix of pixel values, where the two spatial dimensions correspond to height and width, and the third dimension corresponds to the RGB color channels.

These pixel values constitute the fundamental factors of the dataset, as all subsequent analysis steps, including feature extraction, dimensionality reduction, and clustering, will be based on the relationships and variations within these numerical components. No additional metadata or external features are provided, ensuring that all learning emerges directly from the visual information contained in the pixel matrices.

The graphical visualization of the first 100 records shows that this dataset is a collection of different natural photorealistic images, including birds, cats, dogs, and a variety of landscapes with different illumination and depicting diverse seasons.

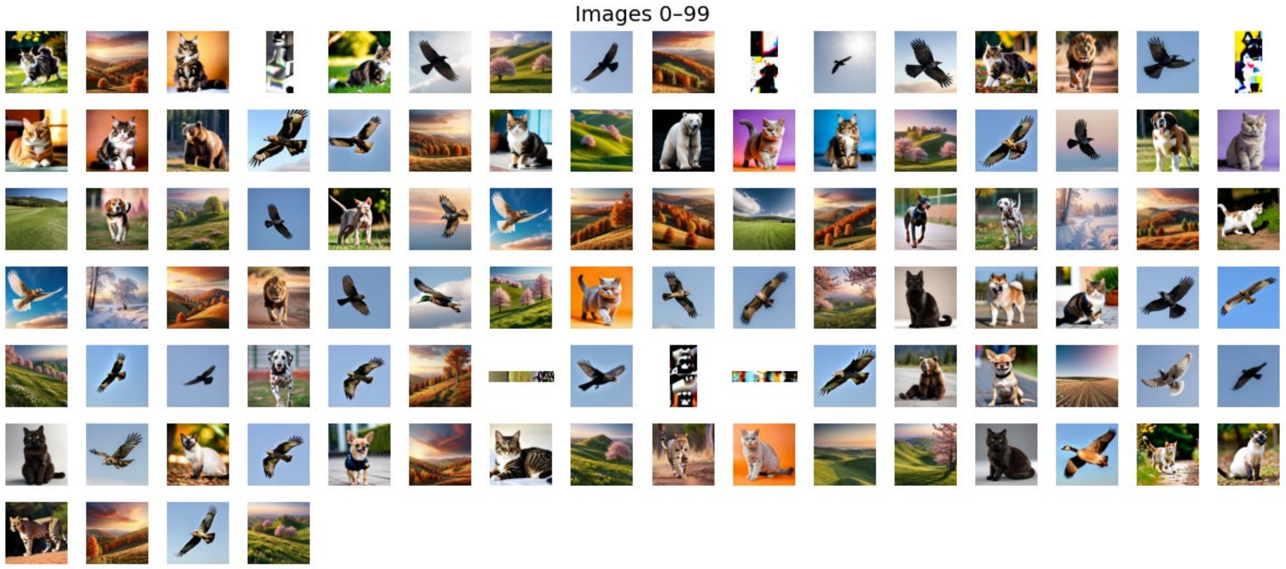


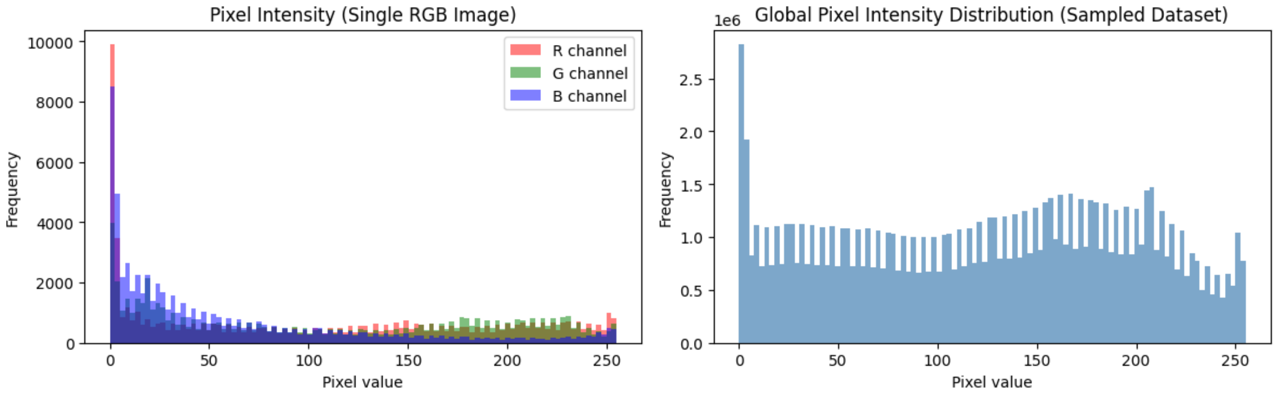
Figure 3.1: first 100 images of the dataset

3.2 Data Cleaning

Firstly, we will remove all the images with resolution different than 256x256, then we will search for duplicates and remove them as well.

```
=== Image Size Filtering Summary ===
Original samples:          577
Kept (correct size):       553
Removed (wrong size):      24
Final X shape:             (553,)
```

```
=== Duplicate Removal (DATASET) ===
Initial samples:  553
Removed:          53
Final samples:    500

SHAPE: (500,)
DTYPE: object
TYPE OF images[0]: <class 'numpy.ndarray'>
SHAPE OF images[0]: (256, 256, 3)
DTYPE OF images[0]: uint8

Pixel value range:
Min: 0
Max: 255
Mean: 123.635
Std: 73.127
```



After the data cleaning phase, the dataset consists of 500 images, each standardized to a 256 × 256 resolution. The analysis of pixel intensities shows values ranging from 0 to 255, with a global mean of 123.635 and a standard deviation of 73.127.

From the histograms of the RGB channels and the global pixel distribution, we can observe a balanced and well-spread color intensity across the three channels, with no saturation or clipping at the extremes. The distributions reveal a natural variability typical of photorealistic images, confirming that the dataset covers both bright and dark regions without dominant bias toward any single-color component.

This suggests that the dataset is visually diverse, properly normalized, and suitable for feature extraction and clustering in the following stages.

3.3 Describe correlations between different factors of the dataset and justify your assumption that they are correlated or not correlated.

In traditional tabular datasets, correlation analysis is typically used to detect redundant features that can be removed because they are linear combinations of others.

In this case, however, **each feature corresponds to a fixed pixel position within a 256×256×3 image matrix.** The relationships among these features are primarily spatial rather than statistical: pixel values are not independent variables but elements of structured visual compositions. Adjacent pixels often share similar values due to color continuity within objects, while distant pixels may vary significantly depending on the spatial layout and content of the image.

For this reason, a standard correlation matrix between pixel intensities would not yield meaningful insights about feature redundancy or independence. The observed dependencies are intrinsic to the geometry of the images rather than to the data distribution itself.

Therefore, direct correlation analysis is not applied in this context. Instead, potential dependencies among features are implicitly captured and reduced through dimensionality reduction techniques such as PCA, NMF, or autoencoders, which are designed to extract compact latent representations that preserve the most relevant visual structure of the dataset.

3.4 Determine if any data needs to be transformed.

Since each record already consists of numerical pixel values, no mandatory transformations are required to make the dataset suitable for analysis.

Nevertheless, certain transformations of the pixel values may be beneficial depending on the modeling approach adopted.

These adjustments can improve numerical consistency and ensure that the representation of the images remains balanced across all features, allowing subsequent algorithms to operate on a coherent and comparable data scale.

3.5 Using your hypothesis, indicate if it's likely that you should transform data

Given that the dataset consists of RGB pixel matrices, no single transformation is universally required in advance.

Normalization or standardization should be applied only when justified by the specific clustering method being employed, since different algorithms react differently to the scale of the input features.

For distance-based methods such as PCA, K-Means, or linkage clustering, scaling procedures can be introduced to maintain numerical coherence and prevent individual channels or intensity ranges from dominating the computation.

In contrast, models that operate directly on raw pixel structures or deep-learning-based feature extractors may not require preliminary transformations.

Where appropriate, additional adjustments such as contrast correction, saturation control, or resolution changes may be introduced later to refine the consistency of the visual data and improve the quality of the extracted features.

3.6 You should determine if your data has outliers or needs to be cleaned in any way.

As shown in the preliminary integrity checks performed in section 3.5, the dataset contains no missing, invalid, or inconsistent values.
All images share the same shape and data type, and no duplicated or corrupted records are present.

Given that each record represents a real image, pixel variations cannot be interpreted as statistical outliers but rather as genuine visual differences among samples.

Therefore, no data removal, interpolation, or substitution has been applied. The dataset can be considered clean and ready for use in the subsequent dimensionality reduction and clustering steps.

3.7 If you believe that specific factors will be more important than others in your analysis, you should mention which and why. You will use this to confirm your intuitions in your final write-up.

In this dataset, the most informative factors are not individual pixels but patterns of pixel intensities that describe shapes and textures within the images.

These latent structures will be identified through matrix decomposition techniques such as PCA and NMF, which extract the underlying components that best represent visual variability.

Their comparison will be performed in the next section to evaluate which representation provides more meaningful features for clustering.

3.8 Balancing the dataset

Considering that unsupervised models benefit from a balanced dataset, the three classes were equalized in size.

The reduction in total samples is a reasonable trade-off, as methods like NMF, autoencoders, and DEC gain stability and clearer cluster structure when the class distribution is even, while PCA is the only method that remains largely unaffected.

```
Images per label:
  Label animals: 235
  Label birds: 114
  Label landscape: 151
```

After balancing:

```
Images shape: (342, 256, 256, 3)
Images per label:
  Label animals: 114
  Label birds: 114
  Label landscape: 114
```

Step 4: Perform Analysis Using Unsupervised Learning Models of Your Choice, Present Discussion, and Conclusions

In this section, we build and compare multiple models to analyze how different unsupervised learning techniques perform when applied to the photorealistic dataset created from scratch. The objective is to understand how each approach extracts latent features, organizes visual information, and forms clusters that reflect meaningful semantic structures within the images.

While the focus of this work remains on unsupervised learning, a simple supervised baseline is also included to provide a reference point and to highlight the practical gap between self-organized and label-driven methods.

The following models are considered:

- **PCA + K-Means:** a linear projection combined with distance-based clustering, serving as a baseline for variance-driven separation.

- **Hierarchical Linkage + NMF:** a hybrid approach that combines non-negative matrix factorization with hierarchical clustering to identify additive and interpretable visual components.

- **Autoencoder:** a neural model trained to reconstruct input images through a compact latent representation, capturing non-linear dependencies among pixels.

- **Deep Embedded Clustering (DEC):** an architecture that jointly optimizes representation learning and clustering to improve latent separability.

- **Supervised Semantic Classifier (baseline):** a reference model trained on a small labeled subset to estimate the achievable upper-bound accuracy and to contextualize the limitations of unsupervised approaches.

This multi-model comparison allows both quantitative and qualitative evaluation of clustering behavior, illustrating how increasing representational power affects the emergence of semantic structure. At the same time, the supervised baseline serves to confirm the internal coherence and overall quality of the dataset itself, showing that the lower accuracy of unsupervised methods reflects the intrinsic challenge of discovering structure without labels rather than any limitation of the data.

Metrics

In this report, clustering quality is evaluated using the following metrics, each capturing a different aspect of agreement between the predicted clusters and the ground-truth categories in the labelled_dataset.

- **ARI: Adjusted Rand Index** measures the similarity between two partitions while correcting for chance. Higher values indicate stronger alignment between clusters and true labels.
- **NMI: Normalized Mutual Information** quantifies how much information about the true labels is preserved by the clustering, scaled between 0 and 1 for comparability.
- **Homogeneity** measures whether each cluster contains samples belonging to a single true class, penalizing mixed-class clusters.
- **Completeness** evaluates whether all samples of a given true class are assigned to the same cluster, penalizing fragmentation.
- **V-Measure:** The harmonic mean of homogeneity and completeness, providing a balanced summary of cluster purity and class cohesion.
- **Optimized accuracy** aligns cluster IDs with true labels using the Hungarian algorithm, estimating the best possible accuracy achievable after resolving the arbitrary label permutations inherent in unsupervised clustering.

In addition to the main metrics, we also examined the confusion matrix after applying the Hungarian alignment. This representation, however, is not always meaningful in unsupervised settings, especially when the learned clusters do not mirror the ground-truth distribution.

For this reason, the evaluation focuses on permutation-invariant metrics described above, which provide a more reliable indication of the underlying partition quality.

4.1 PCA and K-Means

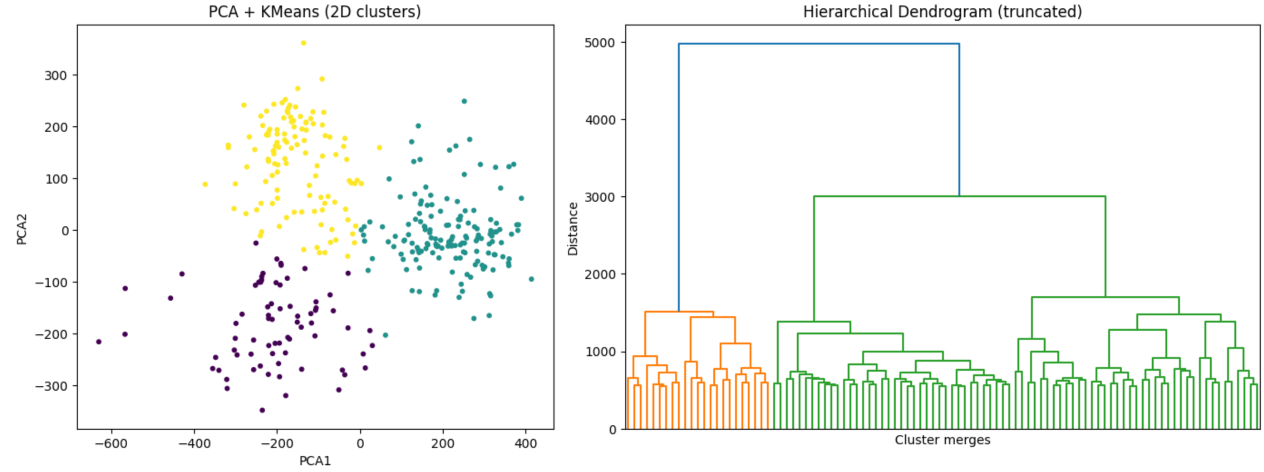
The first model applies Principal Component Analysis (PCA) to reduce each image to a compact set of components that retain most of the variance, summarizing global color and texture while removing redundancy.

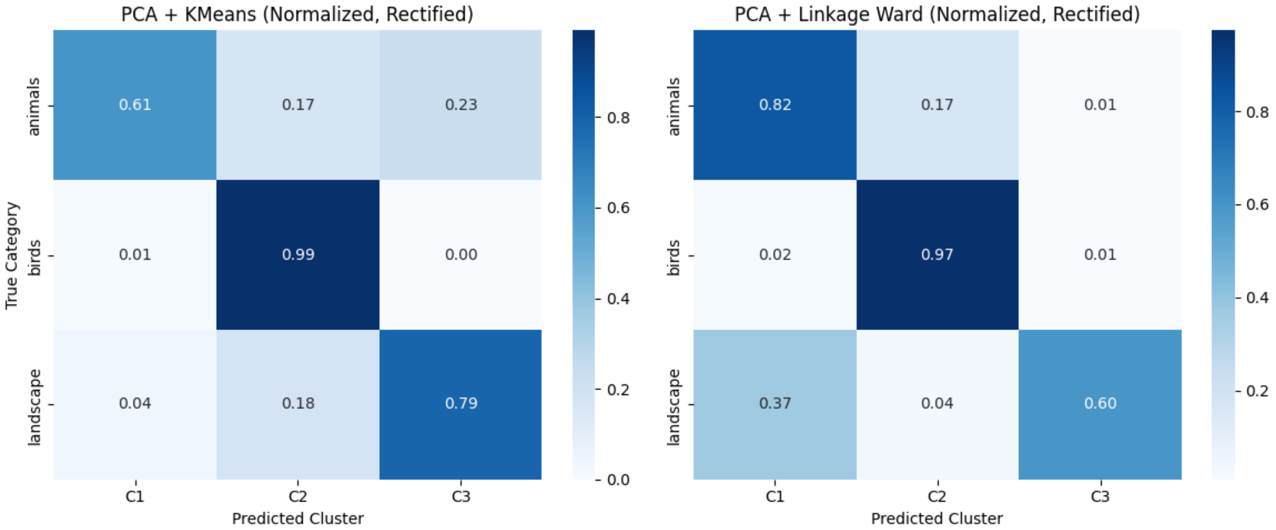
On this representation, we apply K-Means to group images with similar latent features. Because centroid-based methods adapt well to the geometry of PCA-transformed data, K-Means is generally expected to perform better on images, especially when the main structures are captured by the leading components.

As an alternative, we also apply hierarchical linkage clustering (Ward's method) on the same PCA space. Unlike K-Means, linkage builds the cluster structure through irreversible merges that prioritize local variance minimization, offering a more rigid interpretation of the reduced feature space.

Using these two PCA-based variants allows us to compare how flexible centroid-driven optimization and hierarchical merging behave when given the same linear embedding of the images.

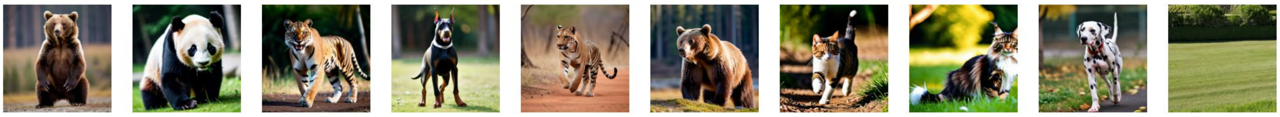
```
PCA reduced to 200 components explaining 95.73% of variance
Silhouette Score: 0.1785
```



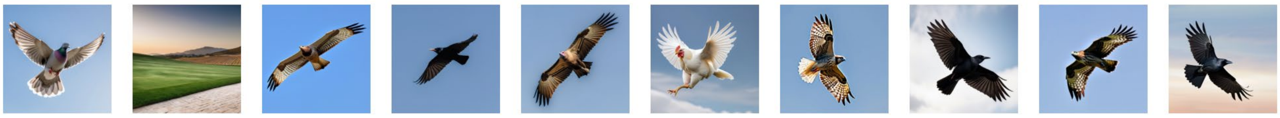


```
PCA K-Means vs Linkage Metrics

K-Means                | Linkage
----------------------+--------
ARI: 0.490             | ARI: 0.525
NMI: 0.487             | NMI: 0.539
Hom: 0.478             | Hom: 0.529
Complt: 0.496          | Complt: 0.549
V-Measure: 0.487       | V-Measure: 0.539
Opt. Accuracy: 0.795   | Opt. Accuracy: 0.798
```

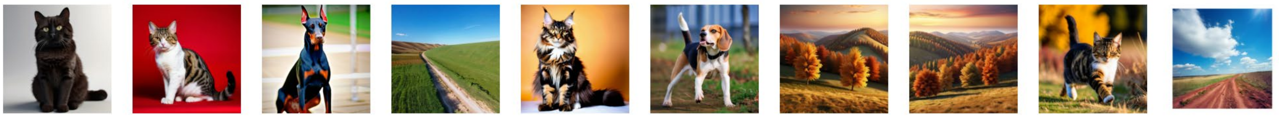
Cluster 0 - showing 10 samples



Cluster 1 - showing 10 samples



Cluster 2 - showing 10 samples



PCA: K-means vs Linkage Comparison Results¶

The comparison of the evaluation metrics for both methods is summarized below:

Metric	PCA K-means	PCA Linkage (Ward)
Adjusted Rand Index (ARI)	0.490	0.525
Normalized Mutual Information (NMI)	0.487	0.539
Homogeneity	0.478	0.529
Completeness	0.496	0.549
V-Measure	0.487	0.539
Optimized Accuracy (Hungarian)	0.795	0.798

As we can see and as expected, K-means adapts well to the geometry of the PCA-transformed space through its centroid-based optimization. It captures the dominant variance directions and maintains a stable separation across the three categories.

Hierarchical linkage behaves differently, but when the class densities are more uniform it benefits from a cleaner spatial layout and produces more consistent boundaries.

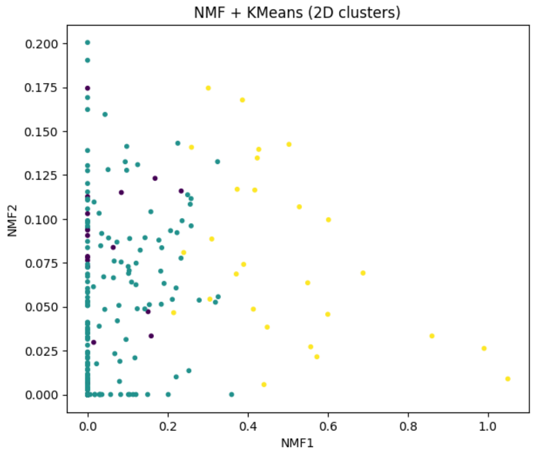
Although both clustering methods produced very comparable accuracy on the ground-truth data, the linkage clustering has a higher ability to preserve coherent boundaries when the class distribution is uniform and the geometric structure is well defined.

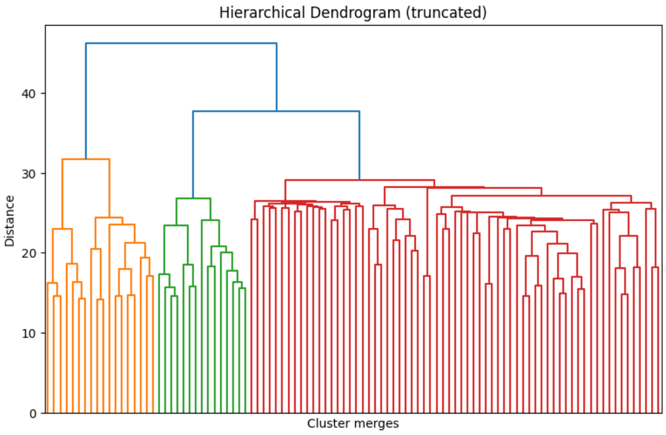
4.2 Non-negative Matrix Factorization (NMF)

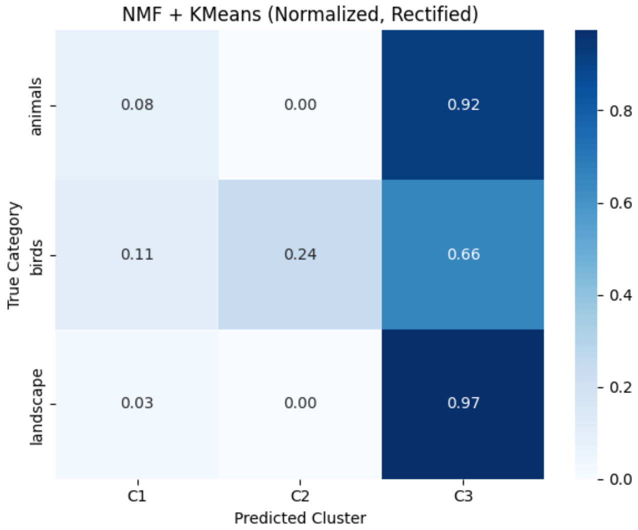
NMF decomposes each image into additive, non-negative components that tend to highlight localized visual patterns such as color regions or texture fragments, offering a parts-based representation that differs substantially from the variance-oriented structure produced by PCA.

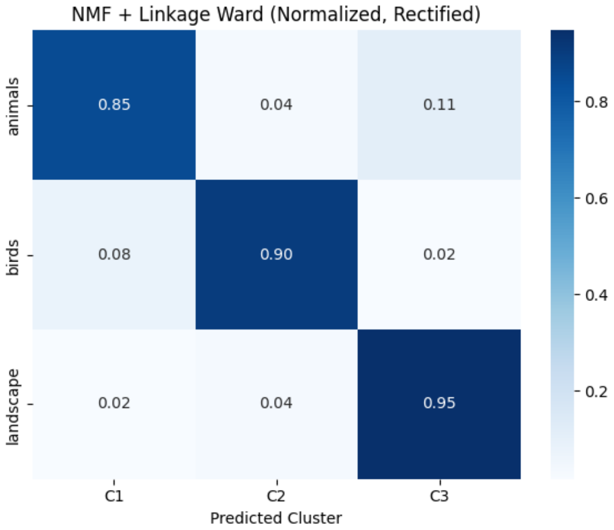
After projecting the dataset into this feature space, we apply both clustering variants used throughout the analysis, allowing us to evaluate how the NMF decomposition influences the formation of groups under the same experimental conditions.

This parallel treatment provides a direct comparison with the PCA-based models and helps determine whether the localized, interpretable features extracted by NMF lead to improvements in semantic organization across the dataset.



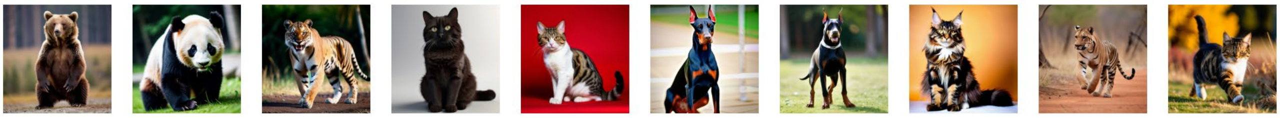




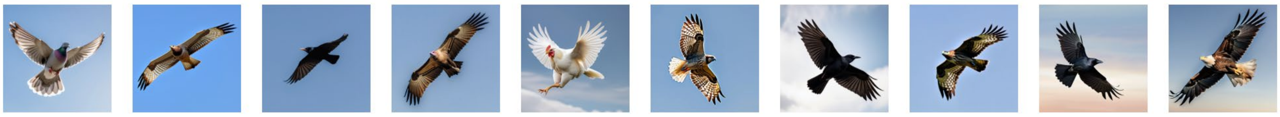


```
NMF K-Means vs Linkage Metrics

K-Means                | Linkage
----------------------+--------
ARI: 0.036             | ARI: 0.726
NMI: 0.132             | NMI: 0.664
Hom: 0.098             | Hom: 0.663
Complt: 0.204          | Complt: 0.664
V-Measure: 0.132       | V-Measure: 0.664
Opt. Accuracy: 0.430   | Opt. Accuracy: 0.901
```

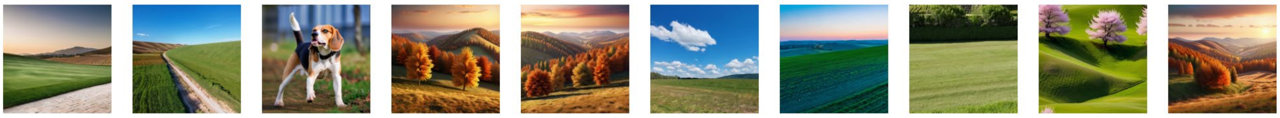
Cluster 0 - showing 10 samples



Cluster 1 - showing 10 samples



Cluster 2 - showing 10 samples



Hyperparameters tuning

Different configurations of n_components, tol, random_state, and initialization methods were tested. The model proved highly sensitive to these parameters, with the most stable and consistent results obtained using 100 components, init='nndsvda', and random_seed = 4.

Results of NMF K-Means vs Linkage (Ward)

Metric	NMF K-means	NMF Linkage (Ward)
Adjusted Rand Index (ARI)	0.036	0.726
Normalized Mutual Information (NMI)	0.132	0.664
Homogeneity	0.098	0.663
Completeness	0.204	0.664
V-Measure	0.132	0.664
Optimized Accuracy (Hungarian)	0.430	0.901

Despite relying on a parts-based decomposition, the behaviour of the two clustering methods diverges sharply. K-Means shows limited agreement with the ground truth and highlights the difficulty of separating the NMF components using centroid-based criteria. The clusters remain diffuse and only loosely connected to the semantic structure of the dataset.

Ward linkage, however, responds very differently once the NMF components are standardized. The hierarchical procedure identifies three compact regions with a level of consistency that is unusually strong for an unsupervised model of this type. All metrics rise substantially, and the optimized accuracy approaches values that are close to supervised performance.

Overall, NMF provides a coherent latent space only when combined with scaling and hierarchical clustering. In this configuration, the results are unexpectedly strong and indicate that Ward linkage is able to capture the underlying structure with a clarity that K-Means does not achieve.

4.3 Convolutional Autoencoder (PyTorch Implementation)

We will implement this method using PyTorch Convolutional Autoencoder.

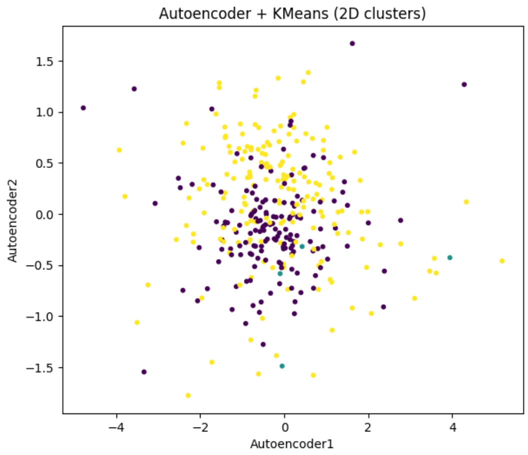
This method differentiates from the previous ones by introducing a non-linear learning process. The architecture consists of an encoder made of convolutional and max-pooling layers that progressively reduce the spatial resolution of the input images, and a decoder using transposed convolutions to reconstruct the original data from a compact latent representation. The network is trained in a fully unsupervised way using a reconstruction loss (MSE), optimized with the Adam algorithm and accelerated through CUDA.

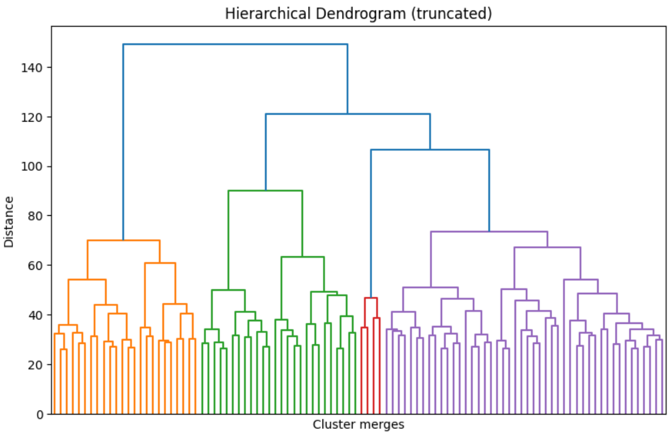
The main operations that will be performed are the following:

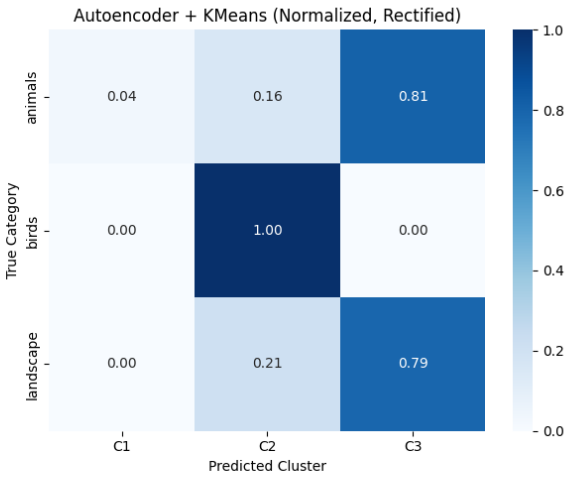
- load the dataset through a custom Dataset class and process it with a DataLoader for batched training, resizing each image to 256×256 pixels and normalizing it to the [0,1] range;
- define a convolutional autoencoder composed of an encoder with convolutional and max-pooling layers, and a decoder with transposed convolutions for image reconstruction;
- train the model using the Mean Squared Error (MSE) loss and the Adam optimizer until reconstruction stability is achieved;
- extract the latent feature vectors from the encoder and apply KMeans clustering to evaluate how well the learned representations capture the intrinsic structure of the dataset.

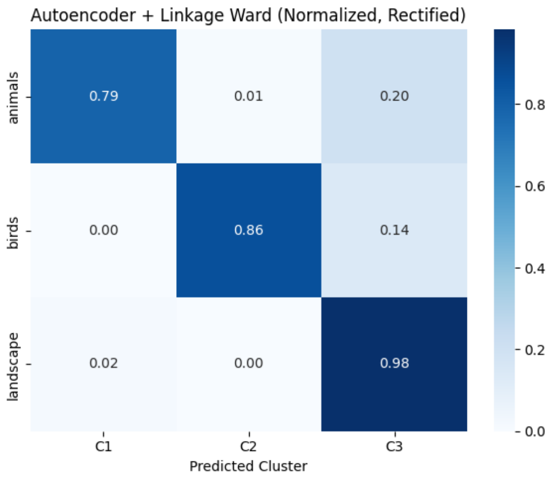
We can expect an improvement in clustering performance, with the ARI increasing by approximately 0.05–0.10 and the overall clustering accuracy reaching around 80–85%.

The model is expected to learn a more coherent latent representation, enabling a clearer separation of semantic categories and a more accurate reconstruction of the underlying visual structures within the dataset.



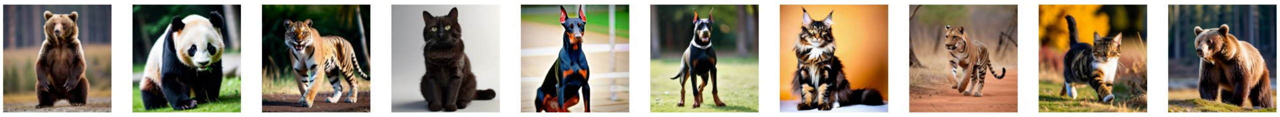




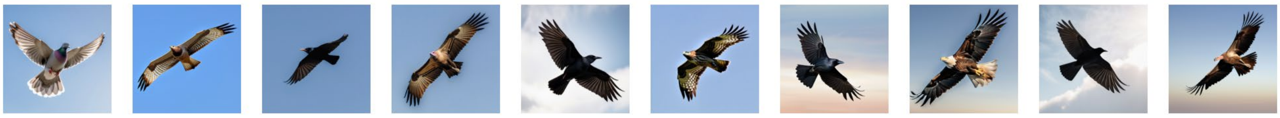


```
Autoencoder K-Means vs Linkage Metrics

K-Means                 | Linkage
------------------------+--------
ARI: 0.387              | ARI: 0.657
NMI: 0.412              | NMI: 0.667
Hom: 0.346              | Hom: 0.660
Complt: 0.510           | Complt: 0.675
V-Measure: 0.412        | V-Measure: 0.667
Opt. Accuracy: 0.608    | Opt. Accuracy: 0.877
```

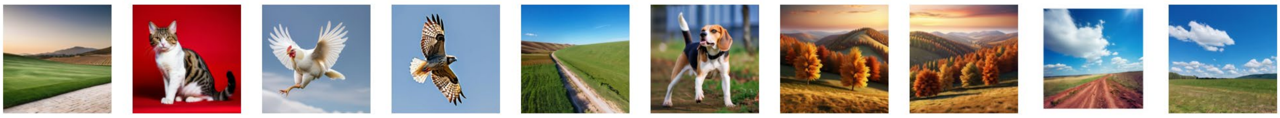
Cluster 0 - showing 10 samples



Cluster 1 - showing 10 samples



Cluster 2 - showing 10 samples



Autoencoder Results

After an extensive series of tests combining seed, batch size, number of epochs, and patience, these metrics represent the best configuration obtained so far:

- seed: 4
- batch size: 8
- max-epochs: 20
- patience: 20

The autoencoder produced a latent space that captures the dominant semantic structure of the dataset. Both clustering methods perform well on this representation, with linkage showing a stronger ability to separate the three visual groups.

The results indicate that this configuration achieves a balanced embedding that supports stable, unsupervised discrimination of the classes.

Metric	AE K-Means	AE Linkage (Ward)
Adjusted Rand Index (ARI)	0.517	0.657
Normalized Mutual Information (NMI)	0.515	0.667
Homogeneity	0.507	0.660
Completeness	0.523	0.675
V-Measure	0.515	0.667
Optimized Accuracy (Hungarian)	0.804	0.877

4.4 Deep Embedded Clustering (DEC)

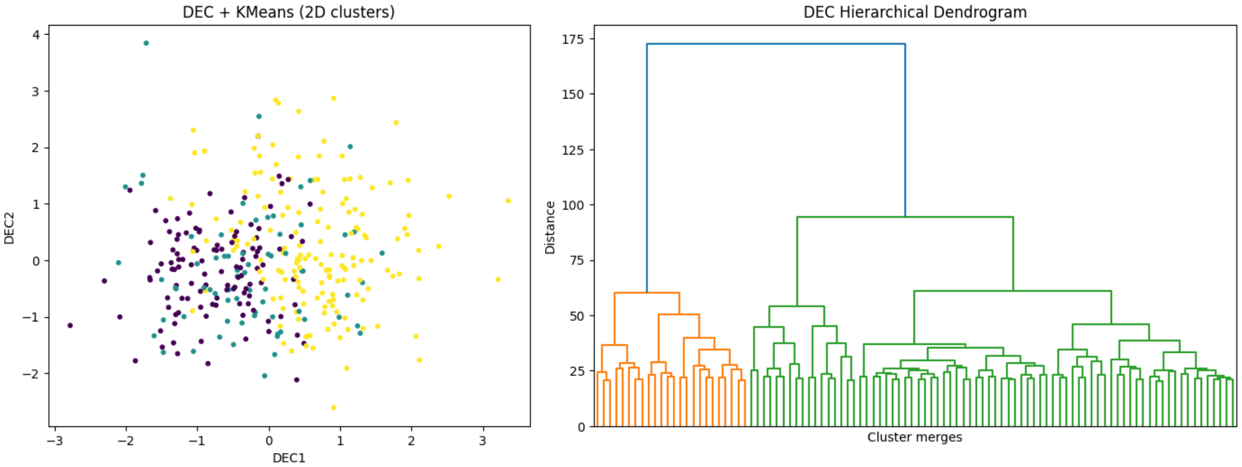
Deep Embedded Clustering is an unsupervised method that integrates non-linear representation learning with a dedicated clustering objective.

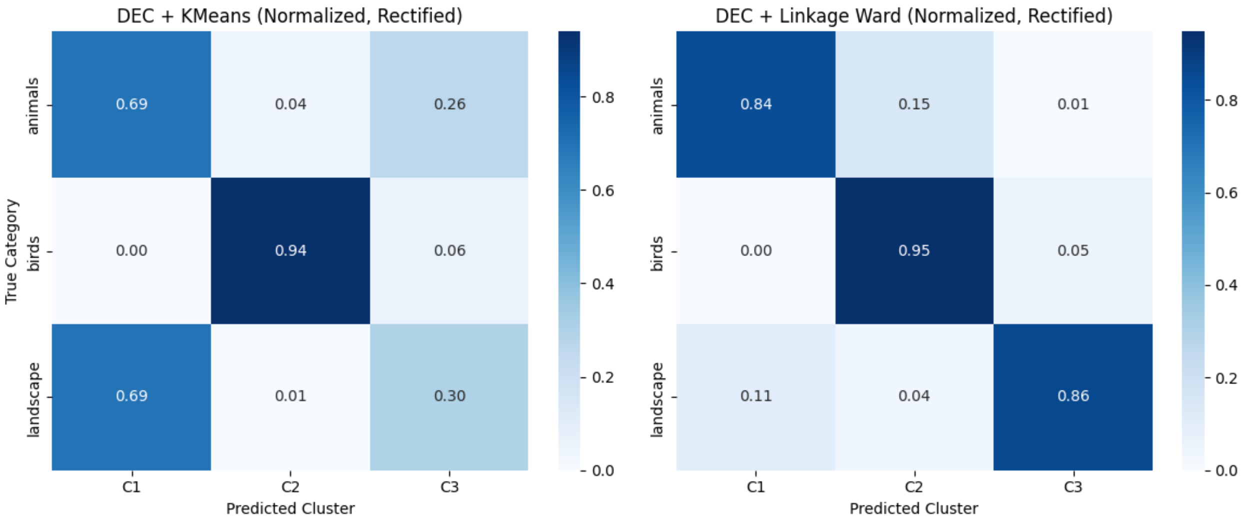
The process begins by training a convolutional autoencoder to construct a compact latent representation that preserves the essential visual structure of the input images. Once a stable reconstruction space is obtained, the decoder is removed and the encoder is further optimized using a clustering-driven loss.

DEC initializes cluster centroids using K-Means in the latent space, then refines both the embeddings and the cluster assignments through an iterative procedure based on the Student's t-distribution. This mechanism sharpens cluster boundaries by increasing the contrast between high-confidence assignments and ambiguous samples, gradually transforming the latent space into a more discriminative representation.

This approach is particularly suitable for datasets such as the present one, where class separation depends on subtle shape regularities rather than simple color statistics.

The combination of non-linear feature extraction and clustering-oriented fine-tuning allows DEC to capture spatial structures that linear methods or purely reconstruction-based encoders tend to attenuate. As a result, DEC is well positioned to uncover meaningful semantic partitions even in moderately complex visual domains.

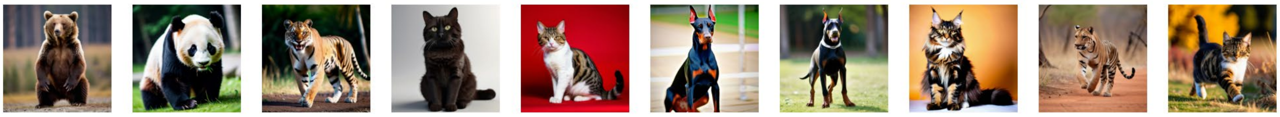




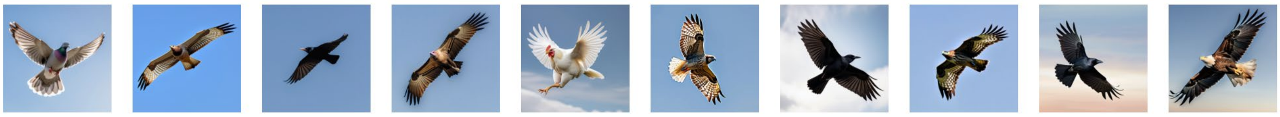
```
DEC K-Means vs Linkage Metrics

K-Means                | Linkage
----------------------+--------
ARI: 0.440             | ARI: 0.684
NMI: 0.471             | NMI: 0.645
Hom: 0.460             | Hom: 0.644
Complt: 0.482          | Complt: 0.646
V-Measure: 0.471       | V-Measure: 0.645
Opt. Accuracy: 0.643   | Opt. Accuracy: 0.883

=== Original dataset class counts ===
birds          : 114
animals        : 114
landscape      : 114
```

```
=== DEC cluster counts ===
Cluster 0: 105
Cluster 1: 129
Cluster 2: 108
```

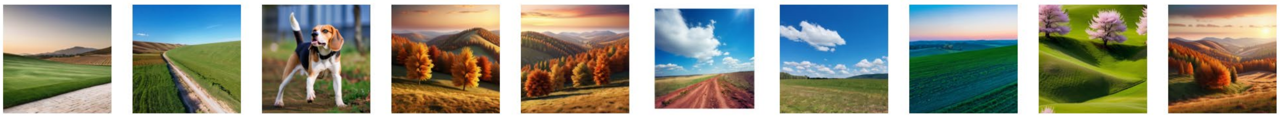
Cluster 0 - showing 10 samples



Cluster 1 - showing 10 samples



Cluster 2 - showing 10 samples



DEC Results

Metric	DEC K-Means	DEC Linkage (Ward)
Adjusted Rand Index (ARI)	0.440	0.684
Normalized Mutual Information (NMI)	0.471	0.645
Homogeneity	0.460	0.644
Completeness	0.482	0.646
V-Measure	0.471	0.645
Optimized Accuracy (Hungarian)	0.643	0.883

As we can see from the above metrics, DEC with K-Means clustering does not provide a meaningful improvement, which is not surprising given that in this particular scenario the RGB images are not easily separable by simple centroid-based boundaries.

Instead, the linkage-based approach delivers a clear performance gain, reaching ARI 0.684, NMI 0.645 and an optimized accuracy of 0.883.

This indicates that the latent space learned by DEC does contain discriminative structure, but its topology is better captured by hierarchical relationships than by fixed-radius partitions.

We can conclude that when the visual differences between classes are subtle and distributed, like in the dataset of images of this work, the linkage clustering appears to exploit the embedding more effectively, producing a separation that is both robust and semantically coherent.

4.5 Supervised baseline (SVC classifier)

Finally, we introduce a supervised SVC model to estimate the maximum accuracy that can be realistically achieved on this dataset.

As the title suggests, this experiment is included purely as a reference point, allowing us to assess the intrinsic limitations of unsupervised learning and quantify the gap between our best clustering results and a label-driven approach.
A Support Vector Classifier is particularly suitable in this setting, as it performs well in high-dimensional continuous feature spaces and has a long record of effectiveness on image data transformed through latent representations or dimensionality reduction.

The aim is not to optimize the supervised model, but to establish a credible upper bound that validates the dataset's separability and frames the performance of the unsupervised techniques in a meaningful way.

Supervised SVC Results

```
Supervised SVC Accuracy: 0.942
Classification Report:
                precision    recall  f1-score   support
animals         0.93      0.89      0.91        28
birds           0.96      0.93      0.95        29
landscape       0.94      1.00      0.97        29

accuracy                            0.94        86
macro avg       0.94      0.94      0.94        86
weighted avg    0.94      0.94      0.94        86
```

A brief comparison with the supervised baseline confirms that the dataset is internally consistent and sufficiently informative, as a simple SVC reaches an accuracy of approximately 0.94 without any specialized optimization.

The lower performance of the unsupervised models was expected, since they operate without labels and rely purely on latent structure.

However, the relatively small gap of about 4% points between the best NMF + linkage configuration (0.901) and the supervised reference (0.94) shows that the unsupervised approach is far from ineffective.

On the contrary, it demonstrates a meaningful capacity to recover semantic groupings directly from the data, without relying on annotated examples, and therefore remains a viable option when labels are costly or unavailable.

Conclusions

In this project we explored a hierarchy of unsupervised representation and clustering strategies on an RGB image dataset with three semantic categories. Starting from basic linear decompositions, and gradually moving toward more advanced neural approaches, we built a clear trajectory:

PCA → NMF → Autoencoder → DEC Fine-Tuned (RGB).

All techniques were applied using only the raw RGB channels as input, and the comparison across K-Means and linkage clustering revealed how different methods extract structure from the same visual signal, highlighting the relationship between latent representation quality, clustering behavior, and computational efficiency.

The following table shows the metrics collected during this work:

Model	Configuration	Accuracy (%)	ARI	NMI	V-Measure	Notes
PCA + K-Means	n_components = 200	79.5	0.490	0.487	0.487	Solid baseline, acceptable performance
PCA + Linkage	n_components = 200	79.8	0.525	0.539	0.539	Slight improvement over K-Means
NMF + K-Means	n_components=100, max_iter=100	43.0	0.036	0.132	0.132	Very poor separation with K-Means
NMF + Linkage	**n_components=100, max_iter=100**	**90.1**	**0.726**	**0.664**	**0.664**	**Highest overall metrics among all models**
Autoencoder + K-Means	epochs=20, patience=20, batch=8	60.8	0.387	0.412	0.412	Moderate results, clearly better than NMF
Autoencoder + Linkage	epochs=20, patience=20, batch=8	87.7	0.657	0.667	0.667	Strong performance, close to supervised
DEC + K-Means	max_epoch=110, patience=30, alpha=1.0	64.3	0.440	0.471	0.471	Better than AE+KMeans, weaker than linkage
DEC + Linkage	max_epoch=110, patience=30, alpha=1.0	88.3	0.684	0.645	0.645	Best model for mobile and GPU implementation
Supervised SVC	C=2, test_size=0.25	94.2				Precision 0.93–0.96, Recall 0.89–1.00, F1 up to 0.97

In summary, the results confirm that even when restricted to raw RGB information, unsupervised learning can recover a meaningful semantic structure from image data.

The progression from PCA to NMF, to neural embeddings with autoencoder, and finally to DEC illustrates how increasingly expressive representations affect cluster separability, with linkage-based methods generally outperforming K-Means across all models.

The results also show that, with targeted tuning and careful selection of parameters and hyperparameters, each method can achieve meaningful performance, with accuracy values ranging approximately from 79% to over 90% depending on the model and clustering strategy.

NMF combined with hierarchical clustering achieved the best overall metrics, demonstrating that simplicity and interpretability do not necessarily limit effectiveness.

Neural approaches, while more costly to train, offered competitive results and showed particular robustness when using linkage clustering, narrowing the gap with the supervised reference model.

From an operational perspective, the choice of model also depends on deployment constraints. While NMF remains the most accurate in absolute terms, DEC provides a compact neural encoder compatible with PyTorch Mobile and GPU execution, making it the most practical option for an Android demonstration where lightweight inference and library support are essential.

Demo

An Android demo app is being developed to demonstrate the practical potential of the DEC-based unsupervised training.

The application uses the device's camera to perform **real-time inference**, detecting whether the object or scene currently in the frame corresponds to one of the three discovered clusters:

- animal,
- bird,
- or landscape.

This lightweight mobile implementation highlights how an unsupervised model, once trained, can be deployed efficiently on edge devices without the need for labeled data or cloud computation.

You can see the demo in action at the end of the video presentation.