Unsupervised Image Clustering Using K-Means and Transfer Learning

Finlay Kolijn
Department of Engineering
Western University
London, ON
fkolijn@uwo.ca

Brendan Gorchinsky
Department of Engineering
Western University
London, ON
bgorchin@uwo.ca

Tiago Alves
Department of Engineering
Western University
London, ON
talves4@uwo.ca

Hayden Moore
Department of Engineering
Western University
London, ON
hmoore44@uwo.ca

***Abstract*— In this project, we developed an unsupervised image clustering system that clusters images of animals based on visual similarities. Our approach integrates feature extraction using transfer learning from a pre-trained convolutional neural network (VGG16) and clustering via the K-means algorithm. We began by preprocessing the images, resizing them and applying normalization to improve model stability and remove noise from the dataset. We then leveraged transfer learning with the VGG16 model to extract features from the images. After extracting feature vectors, the elbow method was used to best estimate the optimal number of clusters. A K-Means model was then trained using the image features, grouping them into clusters defined by the elbow method that ideally corresponds to the five animal categories. Finally, the model's performance is assessed using silhouette scores, which provide insight into how well each image fits within its assigned cluster. Through this project, we demonstrated the effectiveness of combining transfer learning with unsupervised clustering techniques to achieve robust image clustering without the need to label training data.**

***Index Terms* – image clustering, K-means, transfer learning, silhouette scores**

I. Introduction

Image clustering has become a cornerstone of modern computer vision applications, enabling machines to interpret and cluster together similar visual data. One such application is animal species identification, which can support ecological monitoring, biodiversity conservation, and wildlife tracking [1]. Traditional supervised learning techniques, such as Convolutional Neural Networks (CNNs), often require large amounts of labeled data for training, which may not always be feasible or available. In contrast, unsupervised learning methods like **K-Means** clustering offer a viable alternative for pattern recognition in unlabeled datasets [2].

Transfer learning has significantly enhanced image recognition tasks, especially when datasets are small or limited. Transfer learning involves leveraging pre-trained models, typically trained on large-scale datasets like ImageNet, and fine-tuning them for a specific target task [3]. This approach not only reduces computational costs and training time but also improves accuracy by utilizing deep feature representations learned from large amounts of data – see Fig.1.

Silhouette scores are employed as a quantitative metric. The silhouette score measures how similar an object is to its own cluster compared to other clusters, combining concepts of cohesion and separation [4]. It ranges from -1 to 1, where higher values indicate that data points are well matched to their own cluster and poorly matched to neighboring clusters. This score is particularly useful in unsupervised learning scenarios, where traditional accuracy metrics requiring ground truth labels are unavailable.

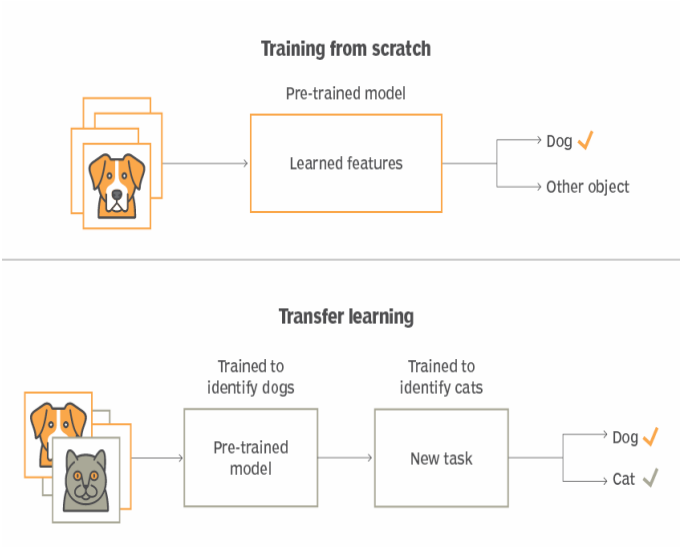


Fig. 1: How transfer learning works.

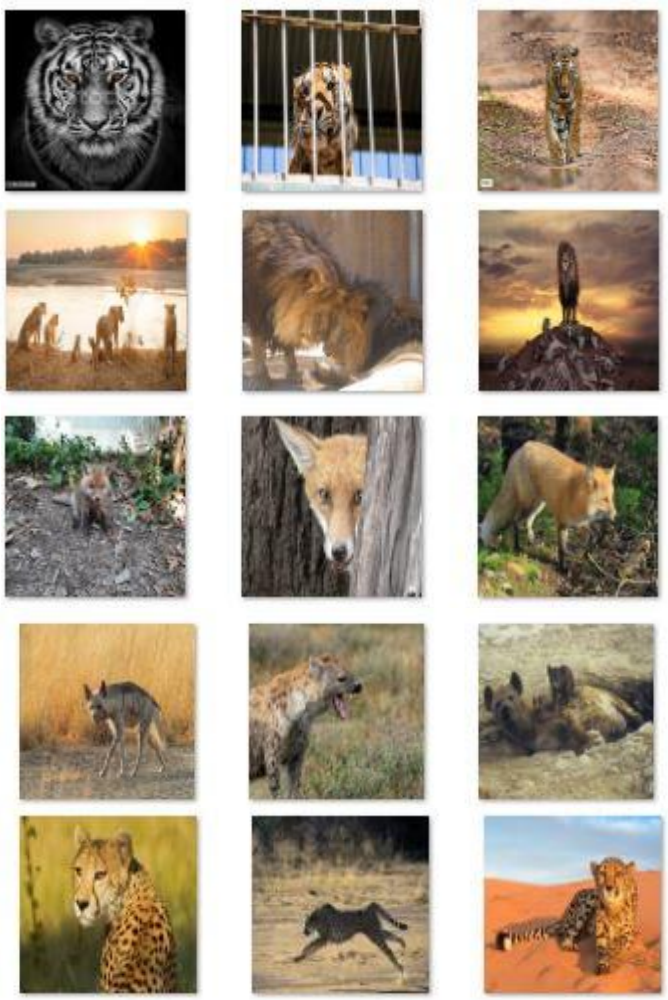


Fig. 2: Sample images from the dataset.

In this paper, we will investigate the unsupervised approach of clustering images of animals using K-means and transfer learning. Initially, we selected a much larger dataset with over 50,000 images but due to GPU constraints with feature extraction, we were not able to train our model. Instead, we used a subset of a wild animal image dataset, containing approximately 1,500 images of 5 unique animals (tigers, lions, foxes, hyenas, cheetahs) – see Fig.2. The images are very diverse with different camera angles, numbers of animals, and background noise. The task we consider is image clustering - grouping images based on visual similarities without using predefined labels. We ultimately predict that the model will be able to form tight clusters of each of the 5 animals with minimal overlapping.

The structure of this report will be comprised of: A comprehensive evaluation of previous work in the field and appropriate background information (Background and Related Work); an explanation of AI methodologies used for preprocessing, feature extraction, clustering, and evaluation coupled with project objectives (Methods); a summary of results (Results); and a conclusion with potential next steps (Conclusions and Future Work).

II. Background and Related Work

In the following section, we will provide a detailed discussion on the background and current work in unsupervised learning, image clustering, and transfer learning. Furthermore, we will highlight the research gap that this paper aims to address.

A. Unsupervised Learning and Clustering

Clustering images is a fundamental task in computer vision and unsupervised learning, where the goal is to group similar images together based on shared features or patterns, without using labels. Traditional image clustering methods, such as hierarchical clustering, rely on handcrafted features to represent images. However, these methods often struggle to capture complex patterns in high-dimensional image data. Recent advancements in deep learning have significantly improved image clustering by leveraging feature representations learned through convolutional neural networks (CNNs), which can capture intricate visual patterns such as object shapes, textures, and spatial relationships [7].

Unsupervised learning has emerged as a powerful tool in machine learning, enabling systems to identify hidden patterns in data without labeled training examples. Clustering, a core technique in unsupervised learning, groups data points based on similarity, with K-means clustering being one of the most widely used algorithms due to its simplicity and efficiency. K-means works by partitioning the data into a predetermined number of clusters, iteratively assigning data points to the nearest centroid and updating the centroids until convergence [6]. In the context of image clustering, unsupervised learning methods have become crucial due to the need for training models with large, unlabeled datasets. While simple, K-means is often criticized for its sensitivity to the initial cluster centroids and its reliance on the predefined number of clusters. Recent research has proposed various enhancements to K-means, such as the use of K-means++ to improve the initialization of centroids [8]. We chose K-means due to its greater simplicity and its faster implementation for large datasets over K-means++.

B. Transfer Learning

Transfer learning is a technique that leverages knowledge gained from one task or domain to improve learning in another related task or domain. This approach has gained significant attention due to its ability to mitigate the challenges posed by limited labeled data and computational resources, which are particularly prevalent in deep learning applications. The core idea behind transfer learning is to use a pre-trained model, typically trained on a large, well-labeled dataset, and adapt it for a different, often smaller, dataset with limited or no labels. This is particularly useful in clustering images, where large datasets like ImageNet contain millions of images across thousands of categories [1]. Models pre-trained on such datasets, such as VGG16, have been shown to learn rich, transferable feature representations that are applicable across different domains. A common approach in transfer learning is feature extraction, where the pre-trained model's learned features are used as input to a simpler model for a specific task

[9]. The success of transfer learning has led to its widespread use in many fields beyond computer vision, including natural language processing (NLP) and speech recognition, where pre-trained models such as BERT and GPT have demonstrated remarkable performance across diverse downstream tasks with minimal task-specific fine-tuning [10].

C. Research Gap

The research gap that this paper aims to address lies in the integration of unsupervised learning, transfer learning, and K-means clustering for more effective image clustering, particularly using datasets where labeled data is not available. Transfer learning, by leveraging pre-trained deep learning models, offers a promising solution by extracting rich feature representations from large-scale datasets. However, while transfer learning has been successfully applied to classification and object detection tasks, its potential for improving unsupervised clustering remains underexplored. The gap in existing research lies in the application of transfer learning for unsupervised image clustering, specifically combining the feature extraction power of pre-trained models with the efficiency of K-means clustering for grouping images without labels.

Current studies in unsupervised learning primarily focus on deep clustering algorithms that simultaneously learn feature representations and cluster assignments, but these methods often require more complex architectures and training processes [7]. On the other hand, using transfer learning with K-means clustering for image data has not been extensively investigated, particularly for clustering images from smaller, specialized datasets. By utilizing the pre-trained VGG16 model to extract high-quality features and then applying K-means clustering to group these features, this approach offers the potential to improve clustering accuracy and efficiency while simplifying the process of feature creation.

III. Methods

This section outlines the technical approach used to implement an unsupervised image clustering system using feature extraction and classical clustering techniques. We employed a pipeline consisting of data preprocessing, feature extraction through VGG16, and clustering using the K-Means algorithm. Each stage is discussed in detail, along with the rationale behind key design choices.

A. Research Objectives

The primary objective of this project is to evaluate the effectiveness of combining transfer learning with unsupervised clustering. More specifically, we aim to determine whether high-level features extracted from a pre-trained convolutional neural network can be used to form meaningful clusters of animal images. Additionally, we aim to assess the quality of this clustering using silhouette scores and visualize the separation of clusters in feature space.

B. Dataset and Preprocessing

In this study, we utilized a custom dataset comprising of approximately 1,500 images of five different animal species. The dataset was stored in a ZIP archive and programmatically extracted. To prepare the data for the VGG16 model, a few preprocessing actions were taken. First, images were converted to RBG, and then BGR to ensure consistency. Then each image was scaled to 224x224 pixels. After that, we used the preprocess_input function that is available from Keras' VGG16 model to normalize the pixel values, which is defined as:

$$x_c' = x_c - \mu_c \quad (1)$$

where μ = [103.939, 116.779, 123.68] is the mean pixel value vector used during VGG16 training on ImageNet [5]. x_c is the raw pixel value for a specific colour channel (ranging from 0 to 255), and x_c' is the normalized pixel value. This step is crucial because the VGG16 model was trained on normalized inputs, so feeding it raw RGB or GBR values would greatly degrade the performance of the model.

C. Feature Extraction with Transfer Learning

To extract features from the images, we leveraged a pretrained convolutional neural network (VGG16) on the custom dataset. Models like VGG16 capture features from images like textures and shapes, which are then used for tasks like clustering. VGG16 is a deep convolutional network trained on the ImageNet dataset, and is composed of convolutional and pooling layers, followed by fully connected (FC) layers. For feature extraction, we used the output of the fully connected layer (FC2), which then produces a 4096-dimensional vector. Formally, given an input image $x \in \mathbb{R}^{224\times224\times3}$, the feature extractor f outputs a vector $f(x) \in \mathbb{R}^d$. These output vectors serve as the input dataset for the unsupervised clustering stage. These feature vectors were used as input to the K-means clustering algorithm, which groups the feature vectors into clusters.

We chose to use VGG16 as it is well-known for its ability to effectively capture hierarchical features from images, such as textures, edges, and shapes, which are crucial for computer vision tasks. Importantly, the ImageNet dataset contains a wide variety of animal classes, including many that are like those in our own dataset. This overlap increases the likelihood that the VGG16 model would already have learned relevant visual features for our tasks. For our approach, we passed all of our images through the VGG16 model to extract high-level features, which can be seen in Fig. 3. We used TensorFlow's Keras API to load the pretrained VGG16 model with ImageNet weights.

Layer (type)	Output Shape	Param #
input_layer (InputLayer)	(None, 224, 224, 3)	0
block1_conv1 (Conv2D)	(None, 224, 224, 64)	1,792
block1_conv2 (Conv2D)	(None, 224, 224, 64)	36,928
block1_pool (MaxPooling2D)	(None, 112, 112, 64)	0
block2_conv1 (Conv2D)	(None, 112, 112, 128)	73,856
block2_conv2 (Conv2D)	(None, 112, 112, 128)	147,584
block2_pool (MaxPooling2D)	(None, 56, 56, 128)	0
block3_conv1 (Conv2D)	(None, 56, 56, 256)	295,168
block3_conv2 (Conv2D)	(None, 56, 56, 256)	590,080
block3_conv3 (Conv2D)	(None, 56, 56, 256)	590,080
block3_pool (MaxPooling2D)	(None, 28, 28, 256)	0
block4_conv1 (Conv2D)	(None, 28, 28, 512)	1,180,160
block4_conv2 (Conv2D)	(None, 28, 28, 512)	2,359,808
block4_conv3 (Conv2D)	(None, 28, 28, 512)	2,359,808
block4_pool (MaxPooling2D)	(None, 14, 14, 512)	0
block5_conv1 (Conv2D)	(None, 14, 14, 512)	2,359,808
block5_conv2 (Conv2D)	(None, 14, 14, 512)	2,359,808
block5_conv3 (Conv2D)	(None, 14, 14, 512)	2,359,808
block5_pool (MaxPooling2D)	(None, 7, 7, 512)	0
flatten (Flatten)	(None, 25088)	0
fc1 (Dense)	(None, 4096)	102,764,544
fc2 (Dense)	(None, 4096)	16,781,312

Fig. 3: Summary of the model built from VGG16 base

D. Clustering using K-Means

To group images, we applied the K-Means clustering algorithm on features extracted from a modified VGG16 model. Instead of using the full VGG16 architecture, we leveraged transfer learning by removing its top layers, only taking the fc2 as output. These layers we removed were replaced by our own custom layers suited for unsupervised feature extraction. Specifically, we truncated the model after 'fc1' (the first fully connected layer) which outputs a 4096-dimensional feature vector for each input image. The 4096-dimensional feature vectors represent high-level image features extracted by the pre-trained convolutional layers of VGG16.

K-Means is a centroid-based clustering algorithm that partitions data into *k* clusters by minimizing the within-cluster sum of squares (WCSS). The WCSS is the sum of the variance between the observations in each cluster. The distance between each observation and the centroid is measured, and the squared difference between them is calculated. Given a set of *n* feature vectors

$$\{x_1, x_2, \ldots, x_n\} \subset \mathbb{R}^d \tag{2}$$

That are of a proper subset of $\mathbb{R}^d$, Eq. 3 seeks to find k centroids that minimize the WCSS. $\mathbb{R}^d$ refers to the d-dimensional Euclidean space where each feature vector encodes learned representations of an image.

$$argmin_{\{\mu_i\}_{j=1}^k} \sum_{i=1}^{n} min_{j\in\{1,\ldots,k\}} ||x_i - \mu_i||^2 \tag{3}$$

We used the KMeans implementation from sklearn.cluster, with a fixed random state for reproducibility and the default K-Means centroid initialization [7].

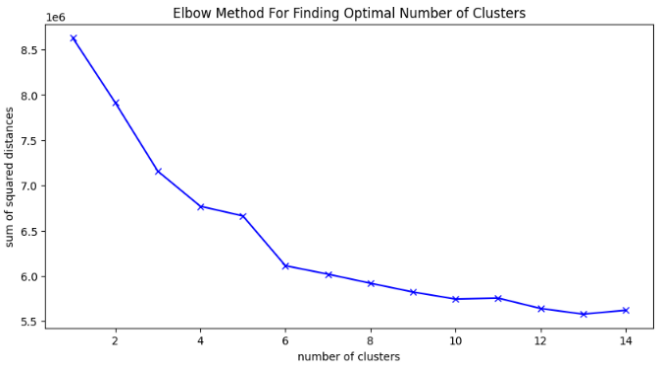


Fig. 4: Elbow Method for finding optimal number of clusters

E. Elbow Method

Although we anticipated five natural clusters due to the corresponding five animal classes, we validated this assumption using the elbow method, which involves plotting the WCSS as a function of *k*, the number of clusters. The optimal *k* is chosen at the point where the marginal gain (i.e. WCSS reduction) sharply decreases:

$$WCSS(k) = \sum_{i=1}^{k} \sum_{x\in C_i} ||x - \mu_i||^2 \tag{4}$$

We tested k ∈ {1,2,...,14}, and we selected the value where the "elbow" in the plot appeared. The optimal number can be visually seen in Fig. 4 where the graph "bends" the most, which was 5 for this dataset. This is how we decided how many clusters we needed.

IV. Results

A. System requirements, architecture, and implementation

The whole project was completed using a python notebook in Google Colab. We utilized the Nvidia Tesla T4 GPU with 16GB of GDDR6 memory and 2560 CUDA cores. We ran into T4 GPU usage limit issues with larger datasets which ultimately led us to use the animal dataset we selected. We also programmed our model to use the CPU if the CUDA GPU limit was reached. The notebook includes a detailed breakdown of all our code including image preprocessing, building the model, extracting features, clustering, and all evaluation methods.

B. Evaluating Performance I: Silhouette Analysis

Silhouette analysis is a technique used to evaluate the quality of clustering by measuring how well each data point fits within its assigned cluster compared to other clusters. In this project, silhouette analysis was applied to the results of K-means clustering to assess the compactness and separation of the image clusters. The silhouette coefficient for each sample is calculated based on how close the sample is to other points in the same cluster and how far the sample is from points in the nearest neighboring cluster. The resulting silhouette score ranges from -1 to 1, where a higher score indicates that the sample is well matched to its own cluster and poorly matched to neighboring clusters. In this study, we obtained an average

score of 0.13 across the 5 clusters. While an average score of 0.13 is not extremely high, the clusters are still very tightly defined. The score can be attributed to the fact that foxes, hyenas, cheetahs, lions, and tigers all share similar structural features, which potentially caused closer together neighbouring clusters. This was one of the reasons we chose this dataset. The similar features of the 5 animals required more demanding clustering. Even though these clusters may be close to one another, the clusters are very tight and mostly include only one animal each. See Fig. 5 for the plot of the individual cluster silhouette scores.

C. Evaluating Performance II: Cluster Animal Distribution

To evaluate the performance of K-means clustering and how tight of clusters it produced, we created pie charts for the animal distribution of each cluster. It is important to note that while we had access to the image names, the model had no access to the names of the files. The way we decided to calculate "accuracy" was by determining the animal with the most appearances within the cluster and dividing this number by the total number

To evaluate the performance of K-means clustering and how tight of clusters it produced, we created pie charts for the animal distribution of each cluster. It is important to note that while we had access to the image names, the model had no access to the names of the files. The way we decided to calculate "accuracy" was by determining the animal with the most appearances within the cluster and dividing this number by the total number of images in the cluster. Cluster 0 had a total of 329 images, with 327 being cheetahs for an accuracy of 99.39%. Cluster 1 had a total of 305 images, with 287 being hyenahs for an accuracy of 94.1%. Cluster 2 had a total of 285 images, with 276 being lions for an accuracy of 96.84%. Cluster 3 had a total of 267 images, with 244 being foxes for an accuracy of 91.39%. Cluster 5 had a total of 274 images, with 267 tigers for an accuracy of 97.45%. On average, the cluster accuracy came out to be 95.83%. While accuracy is not definitively quantifiable for unsupervised learning, as the nature of clustering means there is no true label for the category, we believe our evaluation reflects the clustering intent accurately given that most images are part of a majority in their cluster. This analysis

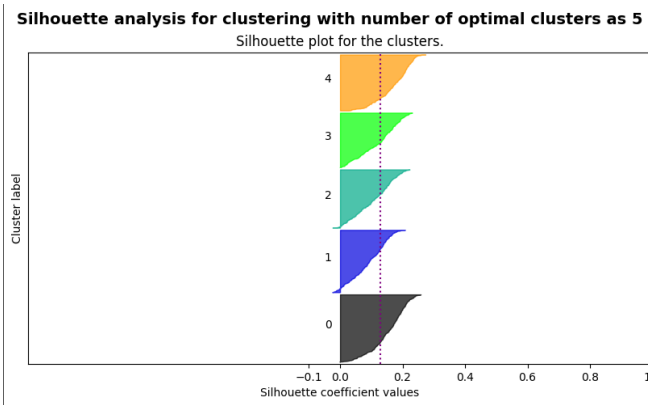


Fig. 5: Silhouette score analysis across the 5 clusters

shows us that while the silhouette score was not close to 1, the clusters were still tightly coupled. The model performed well with separating the 5 different animals See Fig. 6 for an example of a distribution pie chart (the other 4 charts are in the .ipynb file).

D. Evaluating Performance III: Cluster Image Samples

To obtain a more visual representation of the model's performance, we outputted 10 sample images from each cluster. These sample images simply reinforce the animal distributions from the pie charts. See Fig. 7 for the sample images.

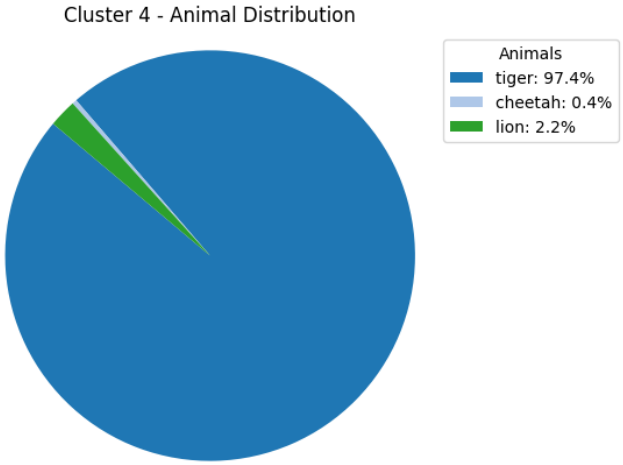
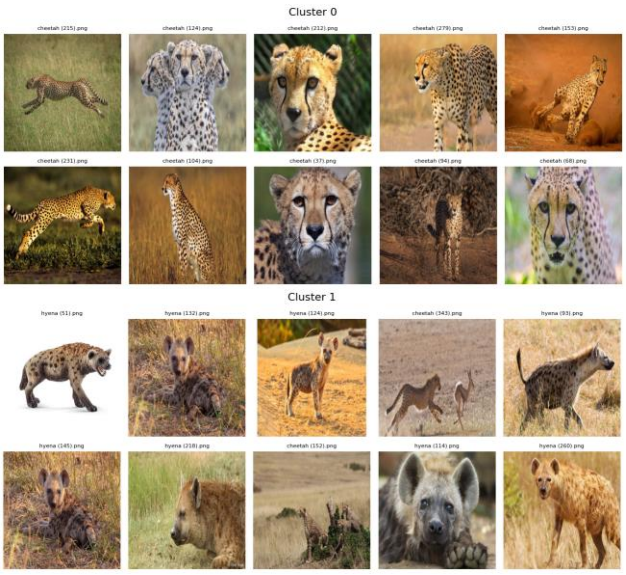


Fig. 6: Cluster 4 animal distribution pie chart



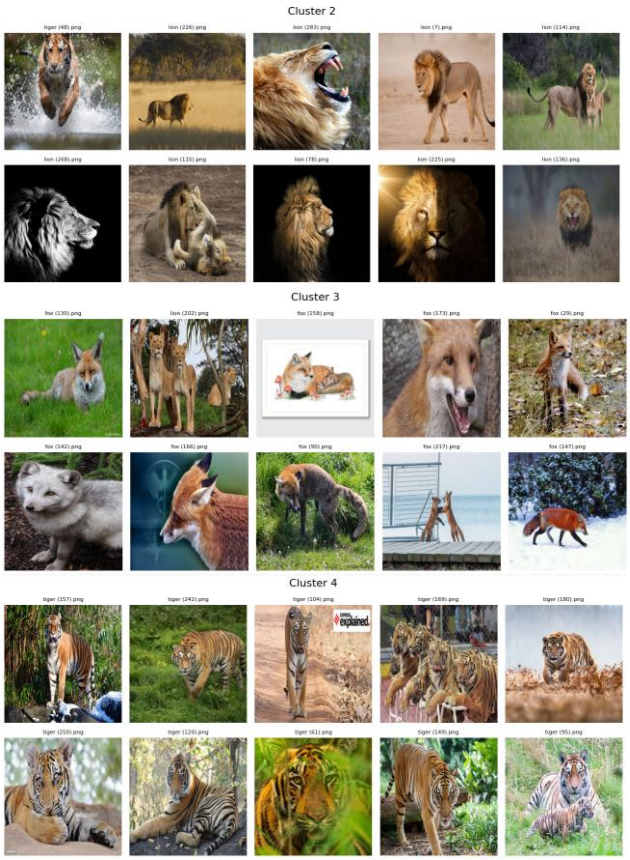


Fig. 7: Sample images from each cluster

V. Conclusions & Future Work

A. *Conclusion*

Overall, the project effectively demonstrated combining transfer learning with unsupervised learning techniques to group images of animals based on visual similarities. By utilizing VGG16 for feature extraction and applying the K-means algorithm for clustering, we were able to successfully sort the images into distinct clusters corresponding to each animal in the dataset. The silhouette analysis showed that while the clusters were tightly defined, some overlap existed between animals that are visually similar. Despite this, the high accuracy within individual clusters, such as cheetahs (99.39%), and tigers (97.45%), showed the clustering process was effective.

B. *Future Work*

One limitation of this work is that the clustering algorithm we used is not the current most efficient algorithm but a relatively simple one. In future iterations, we could try experimenting with other clustering algorithms such as DBSCAN to address this limitation and potentially improve clustering performance.

REFERENCES

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, *"ImageNet classification with deep convolutional neural networks," Advances in Neural Information Processing Systems*, vol. 25, 2012. https://dl.acm.org/doi/10.1145/3065386

[2] J. MacQueen, *"Some methods for classification and analysis of multivariate observations,"* in *Proc. Fifth Berkeley Symp. on Math. Statist. and Prob.*, vol. 1, no. 14, pp. 281–297, 1967. https://www.semanticscholar.org/paper/Some-methods-for-classification-and-analysis-of-MacQueen/ac8ab51a86f1a9ae74dd0e4576d1a019f5e654ed

[3] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 10, pp. 1345–1359, Oct. 2010. https://dl.acm.org/doi/10.1109/tkde.2009.191

[4] P. J. Rousseeuw, "Silhouettes: A graphical aid to the interpretation and validation of cluster analysis," *Journal of Computational and Applied Mathematics*, vol. 20, pp. 53–65, 1987 https://www.sciencedirect.com/science/article/pii/0377042787901257

[5] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," arXiv:1409.1556, 2014. https://arxiv.org/abs/1409.1556

[6] A. K. Jain, "Data Clustering: 50 Years Beyond K-means," *Pattern Recognition Letters*, vol. 31, no. 8, pp. 651–666, 2010. https://www.sciencedirect.com/science/article/abs/pii/S0167865509002323

[7] X. Xie, R. Girshick, and A. Farhadi, "Unsupervised Deep Embedding for Clustering Analysis," in *Proceedings of the 33rd International Conference on Machine Learning*, 2016, pp. 478–487. https://proceedings.mlr.press/v48/xieb16.html

[8] D. Arthur and S. Vassilvitskii, "K-means++: The Advantages of Careful Seeding," in *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, 2007, pp. 1027–1035. https://dl.acm.org/doi/10.5555/1283383.1283494

[9] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, and M. A. Darrell, "Decaf: A Deep Convolutional Activation Feature for Generic Visual Recognition," in *International Conference on Machine Learning*, 2014, pp. 647–655. https://arxiv.org/abs/1310.1531

[10] J. Devlin, M. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," in *Proceedings of NAACL-HLT 2019*, pp. 4171–4186. https://arxiv.org/abs/1810.04805