

Exploring Classification on Autonomously Generated Dataset

by

Sultan Bauyrzhanuly

B.S., Nazarbayev University (2021)

Submitted to the Department of Data Science
in partial fulfillment of the requirements for the degree of

Master of Science in Data Science

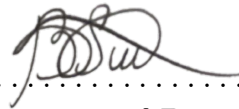
at the

NAZARBAYEV UNIVERSITY

April 2023

© Nazarbayev University 2023. All rights reserved.

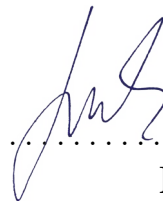
Author



Department of Data Science

7 April 2023

Certified by



Martin Lukac
Associate Professor
Thesis Supervisor

Accepted by

Vassilios D. Tourassis
Dean, School of Engineering and Digital Sciences

Exploring Classification on Autonomously Generated Dataset

by

Sultan Bauyrzhanuly

Submitted to the Department of Data Science
on 7 April 2023, in partial fulfillment of the
requirements for the degree of
Master of Science in Data Science

Abstract

In the era of exponential data growth, the organization and labeling of data play crucial roles. Unsupervised cluster analysis can be utilized to initially group the raw, unlabeled data obtained from a large dataset. This thesis explores the impact of various clustering algorithms, K-Means, DBSCAN, and Gaussian Mixture Models, on the performance of a supervised classification model, specifically AlexNet. The primary objective of the study is to evaluate the classification results on a subset of Places365 dataset after applying different clustering algorithms during the preprocessing phase.

Through a series of experiments, we demonstrate that the choice of clustering algorithm significantly influences the performance of the classification model.

Thesis Supervisor: Martin Lukac

Title: Associate Professor

Acknowledgments

I would like to express my sincere appreciation to my thesis advisor, Professor Martin Lukac, for his priceless mentorship and unwavering motivation throughout each phase of the project. Additionally, I am grateful to my family and friends for their continuous encouragement during this challenging academic year.

Contents

1	Introduction	13
2	Related works	15
2.1	Clustering Algorithms	15
2.1.1	Similarity-Based Clustering	15
2.1.2	Density-Based Clustering	16
2.1.3	Model-Based Clustering	17
2.2	Image Classification	18
2.2.1	Alexnet	18
3	Methodology	21
3.1	Dataset	21
3.2	Experimental Structure	23
3.2.1	Clustering	23
3.2.2	Classification	24
3.2.3	Performance Evaluation	25
3.2.4	Accuracy	25
3.3	Setup	27
3.3.1	Hardware	27
3.3.2	Software and Data	27
4	Experimental results	29
4.1	Clustering	29

4.1.1	K-Means	29
4.1.2	DBSCAN	31
4.1.3	GMM	33
4.2	Classification	35
4.2.1	K-Means	36
4.2.2	DBSCAN	37
4.2.3	GMM	38
4.3	Results comparison	40
5	Conclusion	43

List of Figures

3-1	Image examples from the Places365 dataset	22
3-2	The Experimental Structure of the first and second phase.	22
3-3	The AlexNet architecture for N classes.	25
4-1	10 example images after K-Means on partitions (4 clusters).	30
4-2	10 example images after K-Means on partitions (10 clusters).	31
4-3	Elbow point.	32
4-4	10 example images after DBSCAN on partitions (4 clusters).	33
4-5	10 example images after GMM on partitions (4 clusters).	34
4-6	10 example images after GMM on partitions (10 clusters).	35
4-7	Example images from the dataset. #1 and #2 from train, #3 from test.	36
4-8	Example images after K-Means (top) and after AlexNet on K-Means clusters (bottom).	37
4-9	Example images after DBSCAN (top) and after AlexNet on DBSCAN clusters (bottom).	38
4-10	Example images after GMM (top) and after AlexNet on GMM clusters (bottom).	39

List of Tables

4.1	Image distribution per cluster. K-Means for 4 clusters	29
4.2	Image distribution per cluster. K-Means for 10 clusters	30
4.3	Image distribution per cluster. DBSCAN with eps=13.68	32
4.4	Image distribution per cluster. GMM for 4 clusters	33
4.5	Image distribution per cluster. GMM for 10 clusters	34
4.6	Classification Results. Loss and Accuracy for K-Means during training, validation and test phases.	36
4.7	Classification Results. Loss and Accuracy for DBSCAN during training, validation and test phases.	37
4.8	Classification Results. Loss and Accuracy for GMM during training, validation and test phases.	38
4.9	Classification Results. Accuracy and NLLLoss score for three algorithms during the test phase.	40
4.10	Classification Results. Accuracy by class for three algorithms during the test phase.	40

Chapter 1

Introduction

Machine Learning algorithms have demonstrated impressive performances in different scenarios, image classification, text and speech recognition, and autonomous driving, to name a few [1, 2, 3]. However, these algorithms require large amounts of labeled data in order to train the model and achieve higher accuracy. That data requires time-consuming and expensive processes such as collecting and labeling. Therefore, in recent years, autonomously generated datasets are getting more attention because of their properties of being collected and/or labeled without human intervention [4, 5].

Datasets are referred to as autonomously generated when they are collected and/or labeled by autonomous agents, for example, robots or sensors. With the rise of smart devices and the Internet of Things, the data created autonomously has been growing at an exponential rate [6]. This type of datasets have their advantages over traditionally labeled datasets, which includes the ability of being collected and sorted faster and more efficiently. In such a scenario, clustering algorithms play a crucial role in partitioning unlabeled data.

Clustering data is a common method for preliminary data examination, employed across numerous fields, such as data mining, machine learning, pattern identification, and information retrieval [7, 8, 9]. Clustering is an active area of research, and there are numerous algorithms and techniques available for discovering clusters [10, 11, 12, 13]. Clustering algorithms can generally be categorized into several groups based on their methodologies of how they group data points [8, 14]. This study will be focused

on the following types and algorithms, which will be discussed in the subsequent sections along with particular algorithms:

1. Similarity-based: K-Means
2. Density-based: DBSCAN
3. Model-based: Gaussian Mixture Model

The primary objective of this thesis is to investigate the influence of three distinct clustering techniques on the performance and outcomes of a classification model applied to an autonomously generated dataset. Specifically, this study will analyze the classification results of the widely-used Convolutional Neural Network (CNN) model, AlexNet, after employing various clustering algorithms.

To achieve this objective, the following aspects will be covered in this thesis:

2. **Related works** will provide a brief overview of clustering and classification techniques used in this project.
3. **Methodology** will describe methods used in this thesis for training and validation phases, including the dataset used, the clustering and classification algorithms, and the performance metrics used to evaluate the results.
4. **Experimental results** will present the findings of experiments.
5. In the last section, the study will draw **Conclusions** based on the experimental findings.

This thesis aims to contribute its exploration to the growing body of knowledge on the use of machine learning techniques for autonomously generated datasets. By evaluation of the classification performance of the AlexNet model following the application of three distinct clustering techniques on an autonomously generated dataset, this thesis hopes to provide valuable insights for the future works.

Chapter 2

Related works

2.1 Clustering Algorithms

Clustering is a fundamental task in data mining and machine learning that involves grouping data [14]. It is an unsupervised learning method that involves identifying a finite number of categories, known as clusters, to describe a set of data [15]. Clustering is typically used when the categories are unknown beforehand [15]. Clustering algorithms are used to identify patterns and structures in datasets and are applied in a wide range of fields such as image processing, bioinformatics, social network analysis, and more.

2.1.1 Similarity-Based Clustering

To form clusters, a similarity measure is established between data items and items with similar attributes are grouped together [16]. The process of grouping data into clusters is based on the principle of increasing the similarity within a cluster and decreasing the similarity between clusters [16].

K-Means

K-means is arguably the most prevalent clustering technique in metric spaces and especially among similarity-based algorithms. Similarity-based clustering algorithms

aim to divide a set of data into k clusters in a way that the distributions maximize or minimize a certain criterion function [17]. Every group is characterized by the central point of that particular cluster [14].

Generally, the algorithm's process is started by choosing k cluster centers arbitrarily [16]. After that, k -means subsequently reassigns all data points to their closest centroids and recalculates the centers of the newly formed clusters [16]. This iterative repositioning persists until the objective function reaches convergence. The most frequently employed criterion is the minimization of square error, which is the sum of the squared Euclidean distances between data points and their nearest cluster centroids.

2.1.2 Density-Based Clustering

DBSCAN

Another one of the widely used clustering algorithms is DBSCAN. The Density-based spatial clustering of applications with noise (DBSCAN) is the algorithm based on the idea of density-based clustering, which is suitable for datasets with complex structures, noise, and irregular shapes [18].

Density-based clustering methods are used to group objects into clusters based on their local density rather than their proximity to each other [18]. These methods identify clusters as dense areas separated by low-density regions. They can tolerate noisy data and identify clusters that are not necessarily convex in shape. However, like hierarchical and partitioning methods, density-based techniques struggle to work effectively in high-dimensional spaces because the number of features is usually limited, which makes clustering more difficult.

Many clustering algorithms have been developed in recent decades, but most of them have limitations when it comes to discovering clusters in large spatial databases with noise. In this context, Ester et al. proposed the DBSCAN algorithm, which has proven to be an effective and efficient clustering method [19]. The DBSCAN algorithm works by identifying core points and then expanding clusters around them based on density connectivity [20]. The algorithm has two parameters, the radius

of the neighborhood around each point and the minimum number of points required to form a cluster. The algorithm has been widely used in various fields, including environmental monitoring, bioinformatics, and image processing.

2.1.3 Model-Based Clustering

Model-based clustering is a grouping technique that assumes data is generated from an underlying statistical model [8]. These algorithms are flexible and can adapt to different cluster shapes and sizes, providing a probabilistic interpretation of cluster assignments [8].

Gaussian Mixture Models

Gaussian Mixture Models (GMM) are a versatile and powerful clustering technique based on probabilistic modeling [8]. GMMs have gained popularity in various fields, such as image processing, pattern recognition, and bioinformatics, due to their ability to capture complex data distributions and model clusters with different shapes, sizes, and orientations. GMMs assume that the underlying data distribution is a mixture of multiple Gaussian distributions. Each Gaussian component represents a cluster, characterized by a mean vector and a covariance matrix that define its center, shape, and orientation. In contrast to similarity-based clustering algorithms, the main goal of model-based algorithms is to estimate the model's parameters to best represent the data's structure [8]. So, parameter estimation in GMMs involves finding the optimal values for the means, covariances, and weights of the Gaussian components. The Expectation-Maximization (EM) algorithm is the most common method used for this purpose. The EM algorithm is an iterative process that alternates between two steps: the expectation (E) step, where the posterior probabilities of cluster membership are calculated, and the maximization (M) step, where the means, covariances, and weights are updated based on these probabilities. The algorithm converges to a local maximum of the data likelihood.

2.2 Image Classification

Image classification is a fundamental problem in computer vision, where the goal is to categorize images into predefined classes based on their content [21]. Traditional methods, such as handcrafted feature extraction and machine learning classifiers (e.g., Support Vector Machines), were limited in their ability to handle complex visual patterns and scale to large datasets [22]. The advent of deep learning and CNNs has dramatically improved the performance of image classification tasks [22].

CNNs are a class of deep learning models specifically designed for image processing and recognition tasks [21]. They consist of alternating convolutional and pooling layers that learn hierarchical feature representations from the input data, followed by fully connected layers for classification. At the end of XX century, the convolutional neural network called LeNet was developed and gained a recognition for its successful application in handwritten digit recognition [23]. The last version of this network, known as LeNet-5, was specifically designed for the task of classifying handwritten digits into 10 distinct classes [24]. LeNet-5 utilized a compact architecture that processed images with dimensions of 32×32 pixels. The architecture of LeNet-5 consisted of two convolutional layers, each incorporating 6 and 16 filters of size 5×5 , respectively [24]. Following each convolutional layer, an average pooling layer was applied, and the hyperbolic tangent was employed throughout the network as the activation function. Additionally, LeNet-5 featured three fully connected dense layers, comprising 120, 84, and 10 units, respectively. LeNet-5 achieved an impressive accuracy rate of approximately 99% when evaluated on a separate test dataset. CNNs have been shown to achieve state-of-the-art performance on various image classification benchmarks, outperforming traditional methods by a significant margin [22].

2.2.1 Alexnet

In 2012, the first CNN used at the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) was able to obtain a great success by achieving higher results than the competitors with the top-5 accuracy of 84.7%. This CNN is well-known as AlexNet

was proposed by Krizhevsky, Sutskever, and Hinton [22]. It is a deep CNN architecture that marked a turning point in the field of image classification by demonstrating the potential of deep learning for image recognition tasks. AlexNet consists of eight layers, including five convolutional layers, three fully connected layers, and a final softmax layer for classification. The convolutional layers consist of 3, 96, 256 and 384 filters of size 11x11, 5x5 and 3x3 [22]. This makes the AlexNet a relatively lightweight CNN model compared to more recent models such as VGG and ResNet with 16 layers or more. Key innovations in AlexNet include the use of rectified linear units (ReLU) as activation functions, dropout for regularization, and data augmentation to improve generalization [22].

AlexNet has proven to be effective in learning discriminative features from images [22, 25]. Its deep architecture allows for hierarchical feature extraction, capturing both low-level and high-level visual representations [22]. Following its success in the ILSVRC, AlexNet became a foundational architecture for numerous image classification tasks and spurred further research in deep learning for computer vision. AlexNet has been applied in various domains, such as medical image analysis [25], remote sensing [26], and object detection [27]. The architecture has also inspired the development of more advanced CNN models, such as VGG [28], Inception [29], and ResNet [30].

Over the years, the CNN architecture demonstrated extremely well its capabilities for the image classification. Specifically, AlexNet is a model that was studied extensively and is widely used in research. Additionally, by being a good balance between light and heavy CNN models, the AlexNet makes a well-suited choice as a base evaluation model for better generalization and reproducibility. Clustering algorithms can benefit from feature representations learned by the CNN, by utilizing the pretrained AlexNet as a feature extractor.

Chapter 3

Methodology

3.1 Dataset

In this thesis, a diverse image collection dataset, namely the *Places*, was chosen to facilitate the investigation and evaluation of various algorithms [31]. The specific choice of the dataset is not crucial for the research objectives and findings; rather, it serves as a representative sample to explore the algorithms and methodologies employed in the study.

Places is a large-scale image dataset and there are different variations of the dataset depending on the size of it. For this project, all of the experiments have been done using the images from the Places365-Standard version. The Places365-Standard is a subset of the Places dataset with over 1.8 million images for training [31]. It additionally has 36,500 images for the validation set and similarly 328,500 for the test set [31]. All of the images from the dataset are of good-quality and colored with a resolution of 256x256 pixels [31]. Some of the examples from the dataset could be seen in Figure 3-1. The images were collected from different sources, such as Flickr and Google Image Search [31].

The *Places* dataset is commonly used in the machine learning field. Its diverse collection makes it well-suited for training and evaluating deep learning models for scene recognition and other computer vision tasks as it additionally comes with annotations for the images. The annotations for them were done using a crowd-sourcing



Figure 3-1: Image examples from the Places365 dataset

approach [31]. It has been used in various research studies and competitions, such as the Places Challenge, which is a competition for scene recognition algorithms [31].

For the scale of the project, 145,000 images were randomly selected and downloaded from the dataset. 80% of it was allocated to the training phases and 10% each for the validation and test phases. *Places 365* dataset images undergo a process of random cropping. This is illustrated in the first and second sections of Figure 3-2. For each image, four non-overlapping samples with a resolution of 40x40 pixels are extracted and included in the training set. This random cropping process is also applied to the validation and test sets.

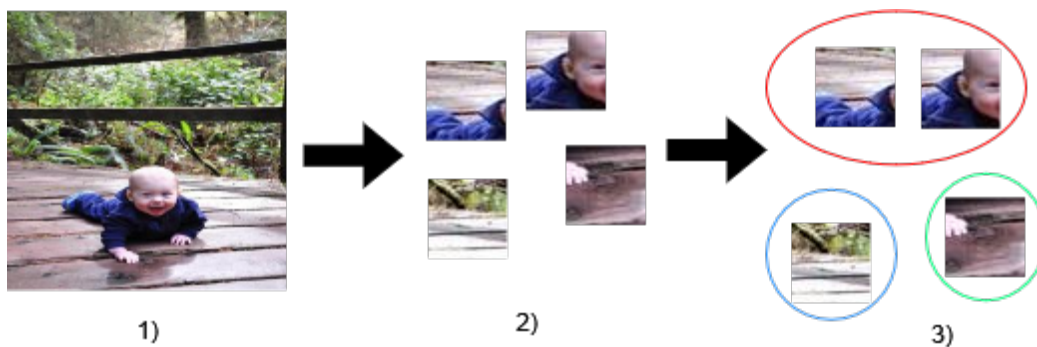


Figure 3-2: The Experimental Structure of the first and second phase.

It is important to mention that the four randomly selected sub-regions of an image always belong to the same dataset partition (training, validation, or test) to ensure consistency and prevent data leakage between these partitions.

3.2 Experimental Structure

The experiments for this project consist of two main parts, clustering and classification.

3.2.1 Clustering

After partitioning the images into smaller pieces, clustering methods are used for the preparation of new datasets, described by the third part of Figure 3-2. During this step, three clustering algorithms have been implemented for grouping up the images into N clusters. For this purpose, the feature values of the images have been used. The feature extraction have been performed using the AlexNet.

The AlexNet was imported with the pre-trained weights on the ImageNet dataset [22] [32]. During the experiments, the images have been passed into the model without crop or size transformations. The features have been extracted after each activation layer of the convolutional layers, meaning after performing ReLu on Conv1-Conv5.

The feature extraction produced a dataframe that contains 1152 low level features per image. After the features have been extracted, they are used in the clustering algorithms mentioned in the Section 2.1.

Next are clustering algorithms' specific parameters that were used during the experiments for this project.

K-Means

K-Means algorithm was implemented using the scikit-learn library [33]. The algorithm was tested using two different values of K, 4 and 10. The "k-means++" was chosen as initialization method because it speeds up the conversion. This method decides initial cluster centroids using sampling based on an empirical probability distribution

of the points' contribution to the overall inertia [33]. The `n_init` parameter was set to 4, so the algorithm would run multiple times with different variations of centroid seeds.

DBSCAN

Differentiating factor of DBSCAN is that during the initialization of the clustering method, the value of epsilon (`eps`) is used rather than the number of clusters. The epsilon is used as a maximum value for two points to be considered neighbours. Therefore, to find the optimal value of `eps`, the elbow method is used.

GMM

The implementation of GMM from the scikit-learn library was used. Similarly to the K-means clustering, the algorithm was tested using two values of clusters, 4 and 10, and "k-means++" as initialization method.

It is important to mention that during the feature extraction and clustering stages, the image sets are combined. However, after the clustering process is completed, the images are separated into their respective clusters and original partitions (training, validation, and test sets). This ensures that the integrity of the dataset divisions is maintained for subsequent analysis and evaluation.

3.2.2 Classification

Before the last phase of the experiments, data transformation is performed to ensure that the input data is in the appropriate format for the CNN model. It was done using the PyTorch library [32]. The transformations applied include resizing, normalization, and data augmentation techniques, which enhance the model's generalization capabilities and reduce the risk of overfitting.

The newly generated data after the clustering and transformation is fed into the classification model, namely AlexNet. For the experiment purposes, the last layer of the model is modified from the initial 1000 classes [22] to the values of `N` used in this

project. The modified representation of the model is shown in the Figure 3-3.

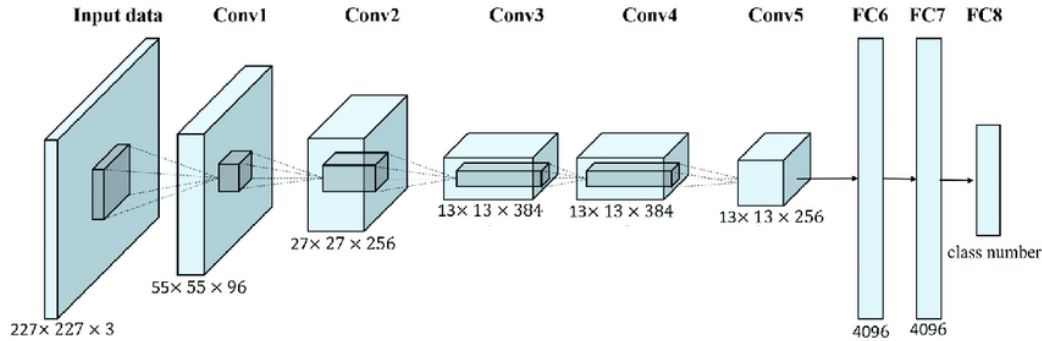


Figure 3-3: The AlexNet architecture for N classes.

Additionally, to get the classification result from the model, the layer of LogSoftmax with the dimension equal to one have been added to the end. During the training of the model, the epoch was set to 50 with a batch size of 32. The Adam optimizer was used with the learning rate of 0.001 [34]. Despite there being dropout layers in the AlexNet model to avoid overfitting [22], additional measure was implemented to avoid that behaviour. In the case when the loss value does not change on the validation set for 10 consecutive epochs, the training is interrupted and the last state of the model with the best performance is saved.

3.2.3 Performance Evaluation

To compare the performance of the mentioned model, two widely used evaluation metrics for classification tasks are employed: classification accuracy and Negative Log-Likelihood Loss (NLLLoss).

3.2.4 Accuracy

Accuracy measures the proportion of correctly classified images out of the total number of images. In this study, the clustering algorithm assigned label was considered as "true" label and the model assigned as "predicted" label. It is a general measure of overall classification performance. It will be evaluated during different phases of the classification and for the each class during the test phase.

Negative Log Likelihood Loss

Negative Log-Likelihood Loss (NLLLoss) is an evaluation metric commonly used in the context of classification problems, especially in the field of deep learning. It measures the dissimilarity between the predicted probability distribution of class labels and the true distribution. In the case of the NLLLoss, a lower score indicates better performance of a model.

$$NLLLoss = -\frac{1}{N} \sum_{i=1}^N \log(p(y_i)) \quad (3.1)$$

In this formula, N represents the number of samples in the dataset, y_i is the true class label for the i -th sample, and $p(y_i)$ is the predicted probability for that class.

In the evaluation of classifiers, various metrics are employed to assess their performance and effectiveness. These metrics provide quantitative measures that aid in understanding how well the classifier performs on a given task. Commonly used metrics include accuracy, precision, recall, F1 score, etc. Among various functions used for classifiers, the NLLLoss has its own advantage when applied to image classification tasks.

By minimizing the negative log-likelihood (equivalent to maximizing the likelihood of the observed data), NLLLoss helps estimate the parameters that best fit the training data, leading to models that capture the underlying patterns and generate predictions that align with the observed data.

NLLLoss is suitable for probabilistic models that assign probabilities to various outcomes like CNN. It is commonly applied in tasks like classification, where the objective is to estimate the probability distribution across different classes. Through the optimization of NLLLoss, the model is incentivized to assign higher probabilities to the correct classes and lower probabilities to the incorrect ones. This characteristic makes NLLLoss a good option for classifiers based on the softmax function.

3.3 Setup

3.3.1 Hardware

All of results mentioned in this paper were obtained during the same "run". Meaning that the clustering algorithms and classification model were prepared and executed on the same machine and hardware. The GPU used in this project is Nvidia RTX 3060.

3.3.2 Software and Data

The described clustering and classification techniques were implemented using Python. The clustering algorithms were imported from the Scikit-Learn library [33], while the AlexNet classification model was downloaded, implemented and modified using the PyTorch library [32].

To guarantee a fair comparison across all three methods, they were each assessed using the same dataset instance. This means that the same distribution of data was used for evaluating the performance of each method, ensuring a consistent and unbiased evaluation process.

Chapter 4

Experimental results

4.1 Clustering

4.1.1 K-Means

Table 4.1 displays the distribution of images across clusters following the application of the K-means clustering algorithm to a dataset with the specified number of clusters set to four. The notable observation here is the relatively balanced allocation of images among the clusters, as no single group appears to be heavily dominant.

Cluster number	Percentage of images per cluster
Cluster 0	28.78%
Cluster 1	21.29%
Cluster 2	32.28%
Cluster 3	17.65%

Table 4.1: Image distribution per cluster. K-Means for 4 clusters

For a better understanding of K-Means clustering results, 10 randomly selected images per cluster have been demonstrated in Figure 4-1. By looking at it, we can form some hypothesises on the "themes" of each cluster. Cluster 1 appears to be characterized by straight line patterns, while Cluster 3 predominantly features leaves, trees, and rounded edges, suggesting a unifying theme of "wavy" lines. Cluster 2 seems to follow a pattern of single blended or blurry color. In contrast to the other

three clusters, Cluster 0 displays a diverse array of images, including building walls, leaves, mountains, and desks, making it challenging to pinpoint a specific theme.



Figure 4-1: 10 example images after K-Means on partitions (4 clusters).

Table 4.2 shows the distribution of images across ten classes following the execution of the K-means clustering algorithm, similar to the previous results. In this scenario, there is a more significant variation in the percentage of images assigned to each cluster.

Cluster number	Percentage of images per cluster
Cluster 0	25.35%
Cluster 1	16.29%
Cluster 2	28.06%
Cluster 3	30.30%
Cluster 4	6.24%
Cluster 5	14.51%
Cluster 6	6.27%
Cluster 7	9.89%
Cluster 8	13.68%
Cluster 9	9.17%

Table 4.2: Image distribution per cluster. K-Means for 10 clusters

Figure 4-2 presents various images from 10 clusters after applying K-Means clustering. We can spot some notable patterns upon closer inspection. Firstly, Cluster 0 seems to concentrate on straight lines, akin to Cluster 1 in Figure 4-1. Cluster 3 likely follows the single-color pattern observed in Cluster 2 of the previous figure. Although

Cluster 5 shares considerable similarities with Cluster 3, it contains distinct examples that feature lines, such as road lanes. While Cluster 8 includes images with grass, Cluster 9 can be considered the "plants" cluster, as it predominantly showcases grass, trees, leaves, and other plant-related elements.

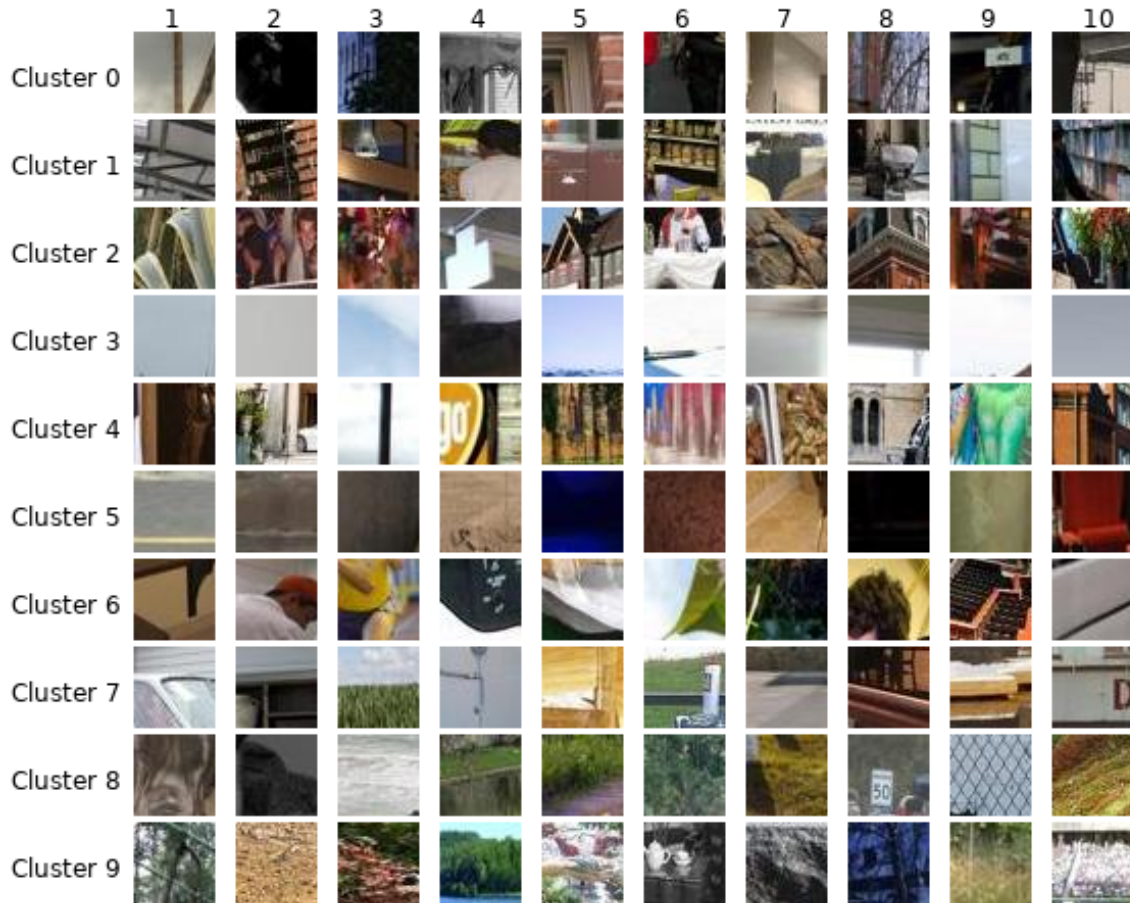


Figure 4-2: 10 example images after K-Means on partitions (10 clusters).

4.1.2 DBSCAN

The plot shown in the Figure 4-3 was obtained by using the NearestNeighbors method from scikit-learn library [33]. By utilizing the elbow method implemented in the KneeLocator Python library, an eps value of 13.68 was calculated.

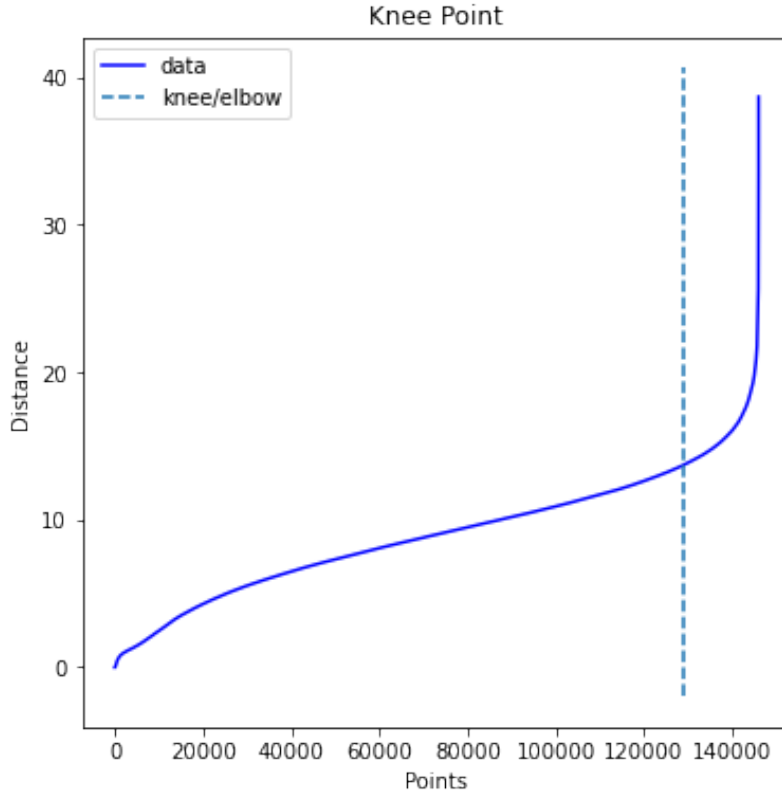


Figure 4-3: Elbow point.

Table 4.3 displays the distribution of images across the three primary clusters identified by the DBSCAN algorithm. Additionally, DBSCAN includes an additional, unique cluster for data that has not been grouped into any specific cluster, which is labeled as noisy or "-1". Upon examining the data distribution, it becomes evident that the algorithm has categorized a significant portion of images into a single category.

Cluster number	Percentage of images per cluster
Cluster -1	11.8%
Cluster 0	78%
Cluster 1	6.1%
Cluster 2	4.1%

Table 4.3: Image distribution per cluster. DBSCAN with eps=13.68



Figure 4-4: 10 example images after DBSCAN on partitions (4 clusters).

Figure 4-4 presents a visualization of 10 randomly selected images from each cluster, enabling a deeper investigation of the underlying patterns. Cluster 0, which is the largest cluster, may have aggregated images featuring various background elements, such as sky, grass, or seats. Cluster 1 shares similarities with Cluster 0 but seems to predominantly focus on uniformly single-colored, blurry images. Lastly, Cluster 2 primarily consists of images containing distinct objects, such as building tips, brake disks, planes, and steel bars.

4.1.3 GMM

Table 4.4 illustrates the allocation of images among clusters. In this case, number of components was set to four. It is another case of relatively balanced distribution.

Cluster number	Percentage of images per cluster
Cluster 0	25.35%
Cluster 1	16.29%
Cluster 2	28.06%
Cluster 3	30.3%

Table 4.4: Image distribution per cluster. GMM for 4 clusters

Similar to the previous clustering algorithms, Figure 4-5 displays discernible visual patterns among the Gaussian Mixture Model (GMM) clustering results. Cluster 3 predominantly consists of single-colored, blurry images. Cluster 2 seems to concentrate on prominent lines, as evidenced by the human head shape (1), chandelier

with its background (3), and bridge (4), among others. Cluster 0 exhibits some resemblance to Cluster 2 concerning "lines"; however, the patterns are not distinctive enough for clear identification.



Figure 4-5: 10 example images after GMM on partitions (4 clusters).

Similarly to K-means clustering results, Table 4.5 also shows the allocation of images with number of components set to ten. Larger variance can be seen here, too.

Cluster number	Percentage of images per cluster
Cluster 0	25.35%
Cluster 1	16.29%
Cluster 2	28.06%
Cluster 3	30.30%
Cluster 4	6.24%
Cluster 5	14.51%
Cluster 6	6.27%
Cluster 7	9.89%
Cluster 8	13.68%
Cluster 9	9.17%

Table 4.5: Image distribution per cluster. GMM for 10 clusters

The examples demonstrated in the Figure 4-6 allow us to see some patterns among the clusters. Firstly, Cluster 5 comprises single-colored blended images, representing the most prevalent cluster type in this project. Cluster 8 shares similarities with Cluster 5 but predominantly includes images containing some form of object. Cluster 0 features images associated with nature, lakes, and coastal buildings. Therefore, a hypothesis can be made that it groups images with the colors and shapes prevalent

in the nature. Cluster 1, on the other hand, likely consists of images emphasizing dividing lines or significant changes in feature values.

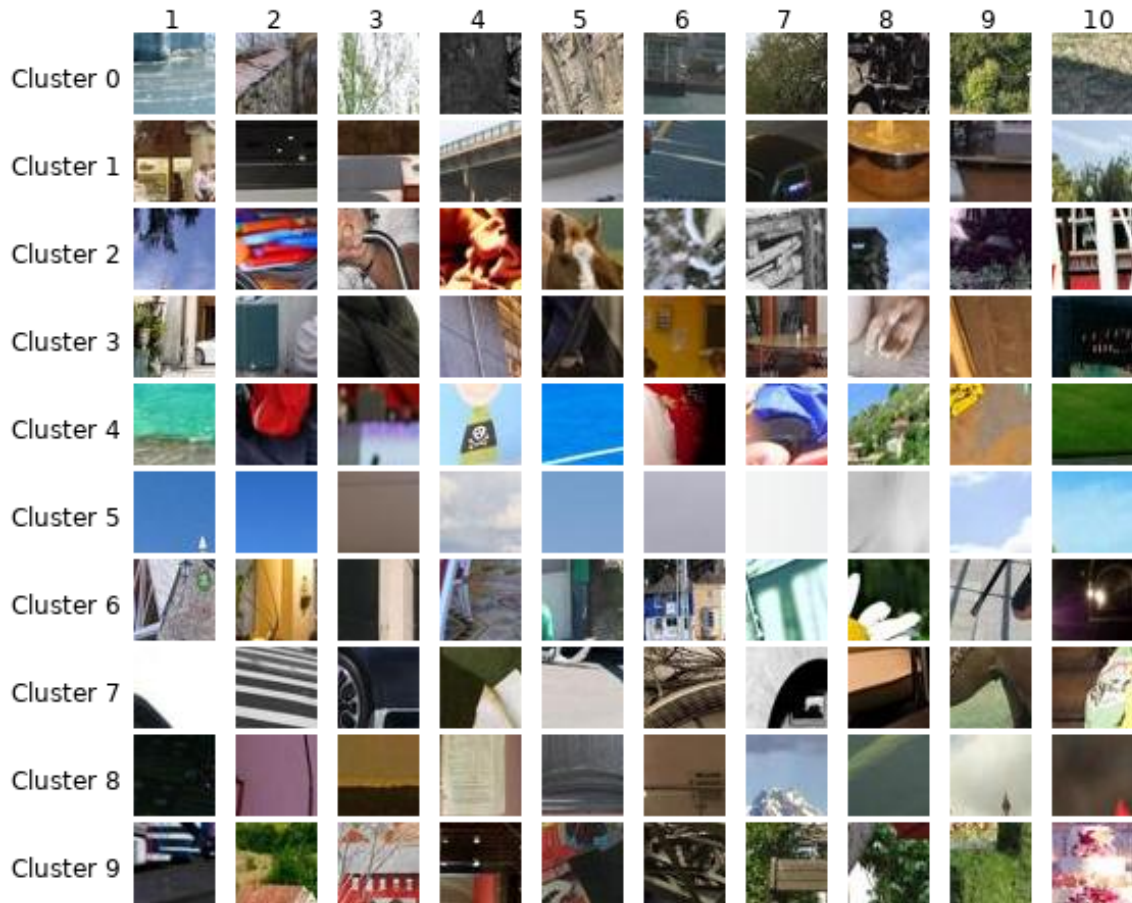


Figure 4-6: 10 example images after GMM on partitions (10 clusters).

4.2 Classification

In this section, multiple experiments were performed to explore the results of the classification model after the application of clustering algorithms. The paper will present three images from the dataset as an example for better understanding of the process. These images shown in the Figure 4-7, were chosen at random, two from training and one from test set.



Figure 4-7: Example images from the dataset. #1 and #2 from train, #3 from test.

4.2.1 K-Means

Table 4.6 presents the accuracy and loss values during various phases of the experiments. In this instance, the AlexNet model was trained on a dataset generated using K-Means clustering, with k-values of four and ten.

Number of clusters	Training loss	Training Acc	Valid loss	Valid Acc	Test loss	Test Acc
4	0.57	77.1%	0.495	79.52%	0.537	78.29%
10	1.102	67.3%	1.054	66.4%	1.209	66.2%

Table 4.6: Classification Results. Loss and Accuracy for K-Means during training, validation and test phases.

The top image in Figure 4-8 presents a selection of results from the K-means clustering algorithm. In the second image, we observe a discernible pattern in Cluster 3, represented by the red color, as it groups together images containing leaves. However, in the first image, the cluster also incorporates a photo predominantly featuring a woman’s hair. Therefore, it is plausible that Cluster 3 encompasses images with wavy lines, as evidenced by the presence of another image in the second example, which similarly showcases a woman’s hair. Moving to the dark magenta colored Cluster 1, it appears to have fewer distinct similarities between elements. Based on the first and second images, it seems to group photos containing straight lines and potentially darker shades. In contrast, Cluster 2, represented by lime-colored outlines, presents three accurate predictions out of three. Following the patterns observed in the other two clusters, Cluster 2 may include uncertain shapes such as clouds or predominantly

feature blue/sky colors. Nevertheless, due to its presence solely in the third image, there are fewer examples available to confirm this hypothesis with certainty.

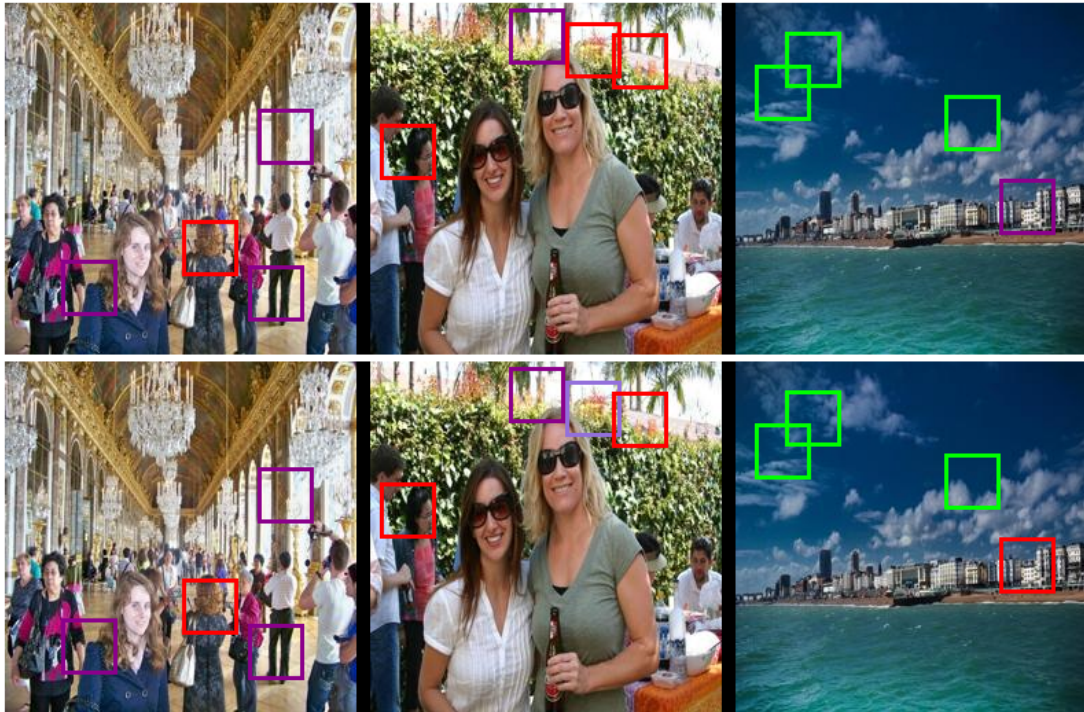


Figure 4-8: Example images after K-Means (top) and after AlexNet on K-Means clusters (bottom).

The bottom image in the Figure 4-8 showcases the same images but after using AlexNet for classifying partitions into classes. For these examples, we can see that it did mostly correct. However it classified leaves on the second and building on the third image incorrectly.

4.2.2 DBSCAN

Number of clusters	Training loss	Training Acc	Valid loss	Valid Acc	Test loss	Test Acc
4	0.17	79.1%	0.25	77.2%	0.22	76.2%

Table 4.7: Classification Results. Loss and Accuracy for DBSCAN during training, validation and test phases.

The classification results presented in Table 4.7 demonstrate relatively high performance, with a classification accuracy of 76.2% on the test dataset.

We can see from the examples in the Figure 4-9 that the clustering algorithm and the model mostly grouped partitions into a single class. The model classified the two "magenta" colored images from the left picture into different clusters.

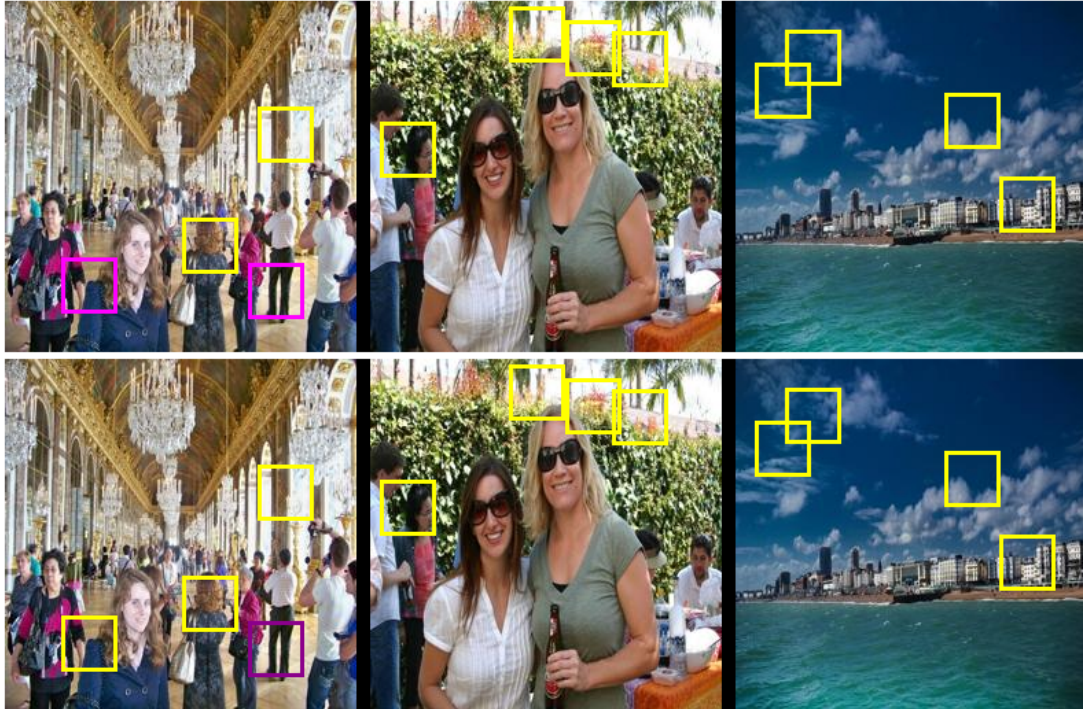


Figure 4-9: Example images after DBSCAN (top) and after AlexNet on DBSCAN clusters (bottom).

4.2.3 GMM

Table 4.8 displays the accuracy and loss values observed throughout different stages of the experiments. In this particular case, the AlexNet model was trained on a dataset created using Gaussian Mixture Model (GMM) clustering, with the number of components set to four and ten.

Number of clusters	Training loss	Training Acc	Valid loss	Valid Acc	Test loss	Test Acc
4	0.70	72.4%	0.636	74.37%	0.621	74.87%
10	0.801	68.2%	0.962	67.9%	0.920	66.8%

Table 4.8: Classification Results. Loss and Accuracy for GMM during training, validation and test phases.

The model demonstrated classification accuracies of 74.87% and 66.8% on the test dataset for the respective configurations.

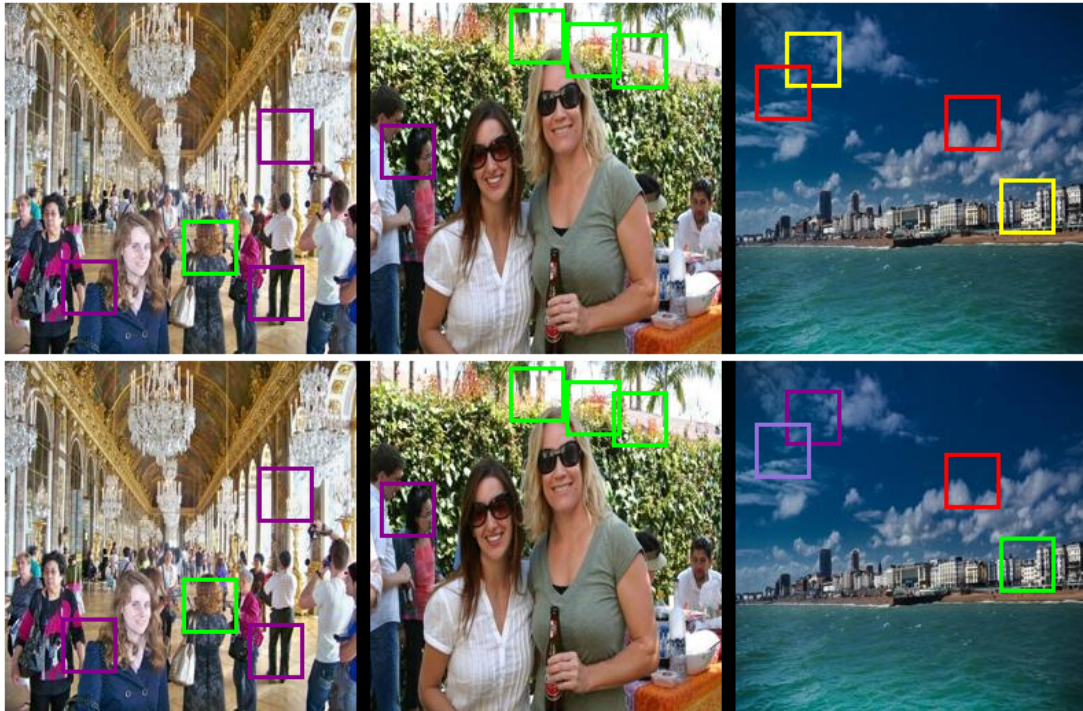


Figure 4-10: Example images after GMM (top) and after AlexNet on GMM clusters (bottom).

The first part of the Figure 4-10 illustrates that the Gaussian Mixture Models algorithm made distinct decisions compared to other clustering techniques. In the third image, the algorithm incorrectly clustered one of the sky images. Nevertheless, it successfully differentiated the building in the third image from other images within the "Dark Magenta" cluster. Furthermore, the algorithm preserved consistency with Cluster 2's "wavy and hair" pattern by incorporating the sample from the upper portion of the second image into the cluster.

This can be contrasted with the second part of Figure 4-10. The clusters are depicted in the left and middle images identically to their counterparts. Therefore, given this results and from Table 4.8, there is high probability of that the model was able to learn some patterns of clusters, and especially of lime and dark magenta colored groups. Nevertheless, the model managed to accurately predict only one out

of the four partitions for the right image. Upon closer inspection, it was able to separate the buildings from the rest, yet into the wrong cluster. The model classified three sky images into three different clusters.

4.3 Results comparison

Clustering Algorithm	4 classes		10 classes	
	Acc	NLLLoss	Acc	NLLLoss
K-Means	78.29%	0.537	66.2%	1.209
DBSCAN	76.2%	0.22	-	-
GMM	74.87%	0.621	66.8%	0.920

Table 4.9: Classification Results. Accuracy and NLLLoss score for three algorithms during the test phase.

Table 4.9 shows the accuracy of the models for N classes when evaluated during a test phase with corresponding clustering algorithm. For the K-Means with 4 classes, the model identified the majority of the patterns and performed well with the NLL-Loss score of 0.537 and the accuracy of 78.29%. The AlexNet demonstrated similar results for the GMM 4-class case with 0.621 score for NLLLoss and 74.87% for the accuracy. This similarity pattern repeats also for the 10 class case. We can see that in some of the example Figures (4-8, 4-10) the substantial amount of the colored boxes match. When it comes to DBSCAN, the model demonstrated a commendable level of accuracy and NLLLoss score on average.

Class number	K-Means	DBSCAN*	GMM
0	65.11%	92.7%	64.64%
1	82.82%	1.1%	55.28%
2	89.38%	0.3%	82.85%
3 (-1*)	73.99%	59.1%	86.78%

Table 4.10: Classification Results. Accuracy by class for three algorithms during the test phase.

However, by looking at Table 4.10, it is essential to acknowledge the enormous variance in accuracy observed when evaluating the accuracy for the individual classes. For instance, when evaluating the Class 0, the model achieved an accuracy of 92.7%

while for the Class 2, the accuracy is a mere 0.3%. The most plausible explanation for this discrepancy is the inability of the model to learn effectively due to the substantial imbalance in class sizes within the dataset, mentioned in Table 4.3. While, the other two cases showcases a relatively balanced accuracies across the classes. It was anticipated that Class 0 for K-Means and Class 1 for GMM would show the lowest results. It is because they are the clusters which contained a diverse collection of images we struggled to assign a specific theme.

Chapter 5

Conclusion

In conclusion, this thesis examined the influence of various clustering algorithms—K-Means, DBSCAN, and Gaussian Mixture Models—on the performance of a supervised classification model, specifically AlexNet. The main goal of this study was to assess the learning insights of AlexNet model in the instance when the labeled data is created through the usage of different clustering methods on a randomly generated dataset.

Through a series of experiments, we found that the choice of clustering algorithm has a substantial impact on the classification model's performance. The models trained on datasets created after the application of Gaussian Mixture Models and K-Means, demonstrated considerably high classification accuracy, while DBSCAN case yielded poorer outcomes due to inadequate clustering of the images. Both, GMM and K-Means, had visually distinguishable patterns in most of their clusters. These findings underscore the importance of selecting the most suitable clustering algorithm for the specific dataset and classification task in question.

For future research, it would be valuable to investigate other clustering algorithms and their effects on various classification models. Additionally, examining the use of ensemble methods that combine the strengths of multiple clustering algorithms could lead to further enhancements in classification accuracy. Finally, exploring the impact of feature extraction and dimensionality reduction techniques on the performance of clustering algorithms would contribute to a more nuanced understanding.

Bibliography

- [1] M. Soysal and E. G. Schmidt, “Machine learning algorithms for accurate flow-based network traffic classification: Evaluation and comparison,” *Performance Evaluation*, vol. 67, no. 6, pp. 451–467, 2010.
- [2] Y. Su, S. Shan, X. Chen, and W. Gao, “Hierarchical ensemble of global and local classifiers for face recognition,” *IEEE Transactions on image processing*, vol. 18, no. 8, pp. 1885–1896, 2009.
- [3] A. Sun and E.-P. Lim, “Hierarchical text classification and evaluation,” in *Proceedings 2001 IEEE International Conference on Data Mining*, pp. 521–528, IEEE, 2001.
- [4] M. Di Cicco, C. Potena, G. Grisetti, and A. Pretto, “Automatic model based dataset generation for fast and accurate crop and weeds detection,” in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5188–5195, 2017.
- [5] S. Kulkarni, S. Chammas, W. Zhu, F. Sha, and E. Ie, “Aquamuse: Automatically generating datasets for query-based multi-document summarization,” *arXiv preprint arXiv:2010.12694*, 2020.
- [6] A. Taherkordi, F. Eliassen, and G. Horn, “From iot big data to iot big services,” in *Proceedings of the Symposium on Applied Computing*, pp. 485–491, 2017.
- [7] R. O. Duda, P. E. Hart, *et al.*, *Pattern classification and scene analysis*, vol. 3. Wiley New York, 1973.
- [8] X. He, D. Cai, Y. Shao, H. Bao, and J. Han, “Laplacian regularized gaussian mixture model for data clustering,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 23, no. 9, pp. 1406–1418, 2010.
- [9] Z. M. Wang, Y. C. Soh, Q. Song, and K. Sim, “Adaptive spatial information-theoretic clustering for image segmentation,” *Pattern Recognition*, vol. 42, no. 9, pp. 2029–2044, 2009.
- [10] H. Gan, N. Sang, R. Huang, X. Tong, and Z. Dan, “Using clustering analysis to improve semi-supervised classification,” *Neurocomputing*, vol. 101, pp. 290–298, 2013.

- [11] G. Karypis, E.-H. Han, and V. Kumar, “Chameleon: Hierarchical clustering using dynamic modeling,” *computer*, vol. 32, no. 8, pp. 68–75, 1999.
- [12] Z. Min and D. Kai-fei, “Improved research to k-means initial cluster centers,” in *2015 Ninth international conference on frontier of computer science and technology*, pp. 349–353, IEEE, 2015.
- [13] F. Murtagh and P. Legendre, “Ward’s hierarchical agglomerative clustering method: which algorithms implement ward’s criterion?,” *Journal of classification*, vol. 31, pp. 274–295, 2014.
- [14] D. Sisodia, L. Singh, S. Sisodia, and K. Saxena, “Clustering techniques: a brief survey of different clustering algorithms,” *International Journal of Latest Trends in Engineering and Technology (IJLTET)*, vol. 1, no. 3, pp. 82–87, 2012.
- [15] J. Han and M. Kamber, *Data mining: Concepts and techniques*. Morgan Kaufmann Publishers, 2006.
- [16] S. A. Elavarasi, J. Akilandeswari, and B. Sathiyabhama, “A survey on partition clustering algorithms,” *International Journal of Enterprise Computing and Business Systems*, vol. 1, no. 1, 2011.
- [17] A. K. Pujari, *Data mining techniques*. Universities press, 2001.
- [18] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, “A density-based algorithm for discovering clusters in large spatial databases with noise,” *Knowledge Discovery and Data Mining (KDD-96)*, 1996.
- [19] U. Fang, J. Li, X. Lu, L. Gao, M. Ali, and Y. Xiang, “Self-supervised cross-iterative clustering for unlabeled plant disease images,” *Neurocomputing*, vol. 456, pp. 36–48, 2021.
- [20] K. Khan, S. U. Rehman, K. Aziz, S. Fong, and S. Sarasvady, “DbSCAN: Past, present and future,” in *The fifth international conference on the applications of digital information and web technologies (ICADIWT 2014)*, pp. 232–238, IEEE, 2014.
- [21] W. Rawat and Z. Wang, “Deep convolutional neural networks for image classification: A comprehensive review,” *Neural computation*, vol. 29, no. 9, pp. 2352–2449, 2017.
- [22] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, 2017.
- [23] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, “Backpropagation applied to handwritten zip code recognition,” *Neural computation*, vol. 1, no. 4, pp. 541–551, 1989.

- [24] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.
- [25] J. Chen, Z. Wan, J. Zhang, W. Li, Y. Chen, Y. Li, and Y. Duan, "Medical image segmentation and reconstruction of prostate tumor based on 3d alexnet," *Computer methods and programs in biomedicine*, vol. 200, p. 105878, 2021.
- [26] X. Han, Y. Zhong, L. Cao, and L. Zhang, "Pre-trained alexnet architecture with pyramid pooling and supervision for high spatial resolution remote sensing image scene classification," *Remote Sensing*, vol. 9, no. 8, p. 848, 2017.
- [27] A. Benali Amjoud and M. Amrouch, "Convolutional neural networks backbones for object detection," in *Image and Signal Processing: 9th International Conference, ICISP 2020, Marrakesh, Morocco, June 4–6, 2020, Proceedings 9*, pp. 282–289, Springer, 2020.
- [28] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.
- [29] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. Alemi, "Inception-v4, inception-resnet and the impact of residual connections on learning," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 31, 2017.
- [30] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.
- [31] B. Zhou, A. Lapedriza, A. Khosla, A. Oliva, and A. Torralba, "Places: A 10 million image database for scene recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017.
- [32] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in pytorch," 2017.
- [33] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.
- [34] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.