UNSUPERVISED IMAGE CLASSIFICATION OF FISH WITHOUT THE INFERENCE OF CLUSTER NUMBER

by

Bhupathiraju Akhilesh Varma

Submitted in partial fulfillment of the requirements
for the degree of Master of Computer Science

at

Dalhousie University
Halifax, Nova Scotia
April 2023

© Copyright by Bhupathiraju Akhilesh Varma, 2023

This thesis is humbly dedicated to my beloved family, whose unwavering love and support have been my constant motivation and inspiration.

Table of Contents

List of Tables

List of Figures

Abstract

This thesis investigates deep learning techniques, particularly unsupervised image classification, for identifying and clustering fish images captured with underwater cameras. In collaboration with Innovasea, the goal is to streamline fish species identification and reduce labor-intensive manual labeling of camera data at the White Rock Dam test site in Nova Scotia, Canada. We developed an unsupervised clustering framework based on the DeepDPM deep learning model. We first reproduced DeepDPM results on several standard datasets. We then integrated ViT MAE embeddings with DeepDPM and applied ESRGAN-based image processing to enhance fish images, which are often blurry and low resolution. These techniques improved clustering accuracy from 30% to 80% for five species of fish. However, using a cluster visualization tool we developed, we observed that fish with similar appearances were clustered together. Our results demonstrate progress towards automating fish species classification and suggest future avenues of research towards this goal.

Acknowledgements

I would like to express my sincere gratitude to Dr. Christopher Whidden for his invaluable advice, support, and encouragement during this project. Since the beginning of this project, he has been an exceptional mentor, always patient and kind. I would like to extend my appreciation to DeepSense and Innovasea for putting their trust in me and giving me the chance to work with them, which was extremely beneficial to my thesis. I am also grateful to Matt from Innovasea for his constant support and confidence in it and for providing the necessary information for this task. I also wish to thank the MITACS Accelerate program for providing the grant that funded our research. Furthermore, I am thankful to the whole faculty and the Computer Science Department for their thoughtful guidance throughout my coursework. Lastly, I am deeply grateful to my family and friends for their unconditional support throughout this academic year.

Chapter 1

Introduction

The monitoring of fish migration is crucial for various reasons, including maintaining healthy fish stocks and monitoring the impact of human activity and pollution on fish [3]. The majority of fish observation is still done manually, either by people fishing [84] or by professionals watching videos and painstakingly counting fish. This process is not only time-consuming and tedious [12] but also highly prone to errors due to human limitations such as fatigue and a limited attention span [23]. Also, manual monitoring can lead to species misidentification, producing false-positive and false-negative errors, which can result in systematic bias in data [76] [70], make it harder to understand the actual status or distribution of species [72] [11] [22], and negatively influence management decisions [39]. Misidentification problems can happen when identifying small, hard-to-see fish [59], new invaders [57], fish underwater or on video [43] [44], or when data are collected by resource users or volunteers [83] [69] [39] [24]. To overcome these limitations, there is a growing need for automated methods that can identify and classify fish species accurately and efficiently.

This is where machine learning techniques may provide a promising solution by automatically grouping fish based on their species, thereby reducing the need for manual counting and increasing the accuracy and efficiency of fish identification and classification. For example an algorithm that can be used to detect patterns in fish behavior and count the fish more accurately than a human observer, even in heavily polluted waters as done in [29]. Another example can be a convolutional neural network as used by Chen et al. [19] could be trained to recognize the distinguishing features of various fish species based on their appearance in photos and videos. Similarly, an automated system based on a deep convolutional neural network was developed by [55] to identify and group fish species and assists marine biologists learn more about the different kinds of fish and where they live. Knausgaard et al. constructed a deep learning system [58] to recognise temperate fish, whereas Shafait et al. created a system [85] that detects and counts fish from underwater films collected in an uncontrolled environment.

Unfortunately, all these still requires vast quantities of expert-labeled training data. According to Kandimalla et al. (2022) [53] and Ayyagari et al. [9], we need 1000-2000 labeled images of each species of interest across multiple environments (2022). As deep learning models are made up of complex neural networks that learn to recognize patterns and make accurate predictions based on the training data. For deep predictive models to work well, it needs many training data labeled by experts. The more labeled training data, the better the model can learn to find relevant features and work well with new data. If there is not enough training data, the model might try to fit the data too well or not learn the patterns in the data, making it bad at handling new data. Due to the diverse range of fish species, it is often difficult to monitor them without human intervention, which can result in a waste of time and resources. These resources could otherwise be put toward more valuable tasks like improving data analysis methods, which would help us learn more about fish populations and how they act, or doing more research on fish populations, which would help us learn more about their current status and needs, and could help us come up with more effective ways to protect them.

Innovasea has created a system based on deep learning that automatically classifies and counts fish by integrating video and sonar data to monitor underwater fish [53]. Equipment has been installed and tested at the White Rock hydropower dam. Even though the system is automated, it must be taught to identify the fish captured on video as salmon, trout, redfish, or bluefish. Supervised learning involves learning a mapping between input data and output labels, using a labeled dataset. In contrast, unsupervised learning involves identifying patterns and structure in the input data, without the use of output labels. Supervised learning is frequently used for classification tasks, whereas unsupervised learning is used for tasks like clustering or dimensionality reduction. To overcome the problem we have, unsupervised clustering may be used which can automatically group fish species with comparable characteristics, which may greatly reduce the time and effort needed to label images of new species or in new environments.

Unsupervised clustering is a more difficult task than supervised clustering because the model must separate clusters based on inherent qualities of the data rather than specified categories. For this reason, unsupervised clustering is not commonly used and, to the best of our knowledge, has not previously been used for fish. However, recent unsupervised models such as SCAN [37] and DeepDPM [79] have shown great potential to cluster common deep learning datasets such as ImageNet [25] which suggests they may be useful for clustering fish species.

Unsupervised clustering falls into two categories: parameterized (like SCAN [37]) and non-parametric (like DeepDPM [79]). With parameterized models, some data-related hypotheses are formed. On the other hand, non-parametric techniques are made up of a group of algorithms that don't make any assumptions about how training data is mapped. This means non-parametric models are more flexible and can capture more complex relationships in the data but may require more data to perform well. When it comes to clustering, which is a technique of grouping similar data points together based on their characteristics, the difference between parametric and non-parametric models in terms of K value is as follows:

Parametric clustering models: Before training a parametric clustering model, the number of clusters (K) is usually set as a fixed parameter; the user must specify the value of K (the approximate number of clusters), and the correct choice of K is always debatable. The model's architecture and hyperparameters limit the number of clusters in parametric models. In these models, the number of clusters that the algorithm will try to find is based on the value of K. If the K value is well estimated, the number of clusters may only sometimes be ideal. One way to solve the problem is to run the clustering algorithm many times with different K values and choose the one that gives the best results.

Non-parametric clustering models: In these models, the number of clusters is not set as a fixed parameter ahead of time. Instead, the algorithm determines the number of clusters based on the data itself. These models can be more flexible than parametric models, as they do not make assumptions about the number of clusters. However, they may require more data to achieve good performance.

Nevertheless, this has several limitations that make it challenging to implement. Executing the method multiple times on massive datasets can be highly challenging, especially in Deep Learning, as training with various alternative cluster sizes may not always be feasible. It is computationally costly to train, requires a great deal of energy, and has a bad environmental influence; it cannot be scaled. Thus, non-parametric techniques such as DeepDPM developed by Ronen et al [79]. come into play; being a non-parametric model, it does not need the user to provide the K value; instead, the model guesses it by adjusting the K value via split-merges of clusters. It also employs a novel amortized inference for Expectation-Maximization (EM) in mixture models [79]. While this technique has demonstrated promising results in managing big datasets and addressing dataset imbalance, we wanted to explore if it is relevant to our particular use case of classifying fish from underwater footage.

Machine learning classification models use sets of input values called 'features' to make classifications. For unsupervised clustering of images we need to convert an image to an embedding of image features such that the image embedding captures relevant differences between the images. Embedding models pre-trained on large image datasets are commonly used to improve the utilization of unlabeled data and achieve better generalization [20]. However, in many cases, better performance can be achieved by fine-tuning a pre-trained embedding on the dataset of interest or by training a new feature embedding model from scratch, but at the expense of increased training time. DeepDPM [79] uses unsupervised pretrained features extractor namely, MoCo [45] for extracting features from the images similar to how SCAN does. In our case, this was not showing much promising results. So, we used ViT-MAE [46] for pretraining and extracting the embeddings from images. This gave a major boost to the clustering results. Detecting the fish species accurately, it is reasonable to assume that high-quality images can produce good quality embeddings during pretraining. However, detecting fish species from low-resolution images captured by a camera can be challenging for deep learning models. Therefore, ESRGAN, a deep learning technique, was used to improve the image quality and subsequently enhance the accuracy of the classification model. Utilizing these techniques, we observed improvements in clustering metrics (NMI, ARI, and accuracy) from 0, 0, and 37 to 75, 74, and 83 for five fish species when comparing the pre-trained MoCo model without enhancement to the enhanced ViT-MAE model trained with a latent dimension of 64 for 400 epochs. Overall, our results demonstrate that DeepDPM has the potential to cluster fish images by species which may reduce the time needed to label new datasets of fish images.

Our contributions include :

- Replicating the results of DeepDPM on the datasets mentioned in their paper.
- Using DeepDPM model on Fish4knowledge dataset to cluster images into different species.
- Used a deep learning technique ESRGAN to enhance the low-quality images present in fish4knowledge dataset to increase performance of the clustering model.
- Applying embeddings obtained from ViT-MAE pretraining instead of MoCo pretraining to achieve much better quality clusters.
- Developing a tool to visualize the clusters created from the model to better understand them. By analyzing the generated clusters, we observed that similar fish species were clustered together, which provides valuable insights into the model's performance and suggests promising directions for future work in automated fish species classification.

Chapter 2

Related Work

Supervised learning is an approach that facilitates the discovery and optimization of a function capable of associating an input with its corresponding output within an input-output pair, commonly referred to as a "training example" [60]. It teaches computers how to solve problems using a set of examples. The goal is to create a function (f) that can understand the relationship between the input and output in these examples and then use it to predict outputs for new, unseen inputs [81].

The two most common types of supervised learning are classification and regression.

Classification is a machine learning technique that predicts categorical outputs by assigning instances to specified classes based on known input values. It can be applied to structured and unstructured datasets and includes binary and multi-label classification types [31].

Regression is a supervised learning technique that identifies correlations between variables and predicts continuous values based on these relationships. It can be categorized into simple linear regression and multiple regression, which includes linear and non-linear types [31].

For the case of fish classification task, we have a set of images (input) and labels (output) that tell us what type of fish is in the image. Instead of manually selecting features of the fish from the images, which can be challenging and time-consuming, it is often better to gather a large dataset of labeled images and use supervised learning to find the function (f) that maps the input to the output. The model learns the relationship between the images (input) and their corresponding species labels (output). Once trained, the model can predict the species of new, unseen fish images and classify them into families or species. This approach simplifies the classification by automating the identification process [81].

Classification of underwater fish images is difficult because of factors like image size, color, texture, and similarities between different species. Since manual classification takes a lot of time and effort, researchers have developed machine learning and deep learning techniques to help with this task [36] [73].

The process of fish classification typically involves three main steps: preprocessing, feature extraction, and classification. Different techniques and algorithms are used in each step [7]. In preprocessing, the fish images are resized, detected, and cropped. Feature extraction involves extracting image characteristics like shape, size, texture, and color. Different methods are used, like measuring distances and angles, GLCM, Gabor filters, SIFT, and SURF [7].

Some methods used for classification include measuring the fish's length, width, and thickness with laser light [87] and looking at their shape, texture, and color [86] [74]. However, variations in water conditions, background, and similarities between species make classification difficult [73]. Different techniques have been developed to classify fish based on their shape, texture, biomass content, and other physical features. These include algorithms like Support Vector Machine (SVM), Back Propagation (BP) algorithm, hybrid algorithms like HGAGD-BPC, GAILS-BPC, Bayesian classifier, and Convolutional Neural Networks (CNNs) [7].

These methods have varying levels of accuracy, with some reaching up to 100% [73] [7]. Some techniques are less time-consuming and cheaper, like those using a mix of feature extraction and clustering algorithms. For instance the paper proposed by Sarika [82] provided a hybrid CNN architecture for recognizing underwater fish species using the Fish4Knowledge [4] dataset, tackling the problems of low-quality images and complex backgrounds. The system used CNN to find features and SVM and K-Nearest Neighbor (k-NN) to classify them. This gave it better results than traditional and existing deep learning methods at the time. In addition, CNN and other deep learning models like AlexNet, ResNet-152, VGGNet, and YOLO have been used for fish classification, with some achieving high accuracy rates. Real-time applications like Mask R-CNN and GOTURN have also been developed [73].

Unsupervised learning is an approach that focuses on identifying patterns, structures, or relationships within unlabelled data without relying on pre-defined input-output pairs. It enables computers to analyze and process data without being explicitly programmed, allowing the model to discover hidden structures within the data.

The goal is to create a function (f) that can identify these patterns and group similar inputs together, even when no explicit output is provided [52]. Clustering, Principal Component Analysis, Anomaly Detection, and Autoencoders are the four different types of unsupervised tasks that may be performed [52]. Clustering might be the most important unsupervised learning task. It involves finding a pattern in a data set that is not labeled. Hence, a cluster is a group of items that are ”similar” to one another but ”dissimilar” to the objects in other clusters [52] [32].

Clustering can be broadly categorized into two types: parametric and nonparametric [42]. Parametric clustering methods assume a fixed number of clusters (K), while nonparametric methods do not assume a fixed number of clusters and can adapt to the data structure. Some examples of parametric clustering algorithms include k-means, Gaussian mixture models, and hierarchical clustering. Examples of nonparametric clustering algorithms include DBSCAN and mean-shift. The Dirichlet Process Mixture (DPM) model is another good example of Bayesian nonparametric (BNP) clustering.

Recently, deep clustering models, which leverages deep neural networks to learn clustering-friendly representations, has been widely applied in various clustering tasks [99]. These methods can also be broadly classified into parametric and nonparametric approaches. Parametric deep clustering methods can be divided into two types: two-step approaches and end-to-end methods. In two-step approaches, clustering is performed on features extracted in a pretext task, such as running K-means on embeddings transformed by UMAP [71]. Examples include the work of McConville et al. and SCAN [90], which employs unsupervised pre-trained feature extractors. On the other hand, end-to-end deep clustering methods learn both features and clustering simultaneously, often through alternation, using autoencoders [49] or variational autoencoders (VAEs) [56] with an extra clustering loss.

SCAN clustering is a two-step procedure that separates feature learning and clustering. In the first stage, a self-supervised task employs an unsupervised contrastive learning method to obtain semantically meaningful features. This process involves training a deep neural network using a pretext task to facilitate learning useful features from unlabeled data. The authors of SCAN utilize either MoCo [45] or SimCLR [16] for feature representation learning.

In the second stage, the clustering task, SCAN [90] performs clustering on the learned feature representations using the K-means algorithm. The model is fine-tuned with a clustering goal that makes it easier for meaningful groups to form. The objective function comprises two main components: (i) maximizing the similarity between each data point and its cluster centroid and (ii) promoting uniformity across clusters via maximization of the entropy of the cluster assignments [90].

In the absence of ground-truth annotations, SCAN [90] is employed to group images into clusters that make semantic sense automatically. SCAN assumes a fixed number of clusters (K) and uniform class weights, implying a balanced dataset assumption. This assumption may be unrealistic in purely unsupervised scenarios, and SCAN's performance deteriorates when the estimate of K is inaccurate. Despite these limitations, SCAN achieves state-of-the-art (SOTA) results in unsupervised image classification tasks [90].

Nonparametric deep clustering methods combine deep learning and nonparametric clustering techniques to find the optimal number of clusters. Some methods, like AdapVAE [100], use a DPM prior [8] [35] [8] for a VAE [56], while others, like DCC, use a nearest-neighbor graph in the latent space of an AE [49] [79].

DeepDPM [79] is a state-of-the-art (SOTA) nonparametric deep clustering method that figures out the number of clusters (K) as it learns, so it does not need to know K ahead of time. DeepDPM used two ways to extract features: an end-to-end method where features and clustering are learned using alternative optimization and a two-step method where features are learned once before clustering and kept the same. The two-step technique used SCAN [90] and MoCo [45] for unsupervised feature extraction. In end-to-end, similarly to DCN [97], an autoencoder is trained with a reconstruction loss. The method beats both deep and non-deep nonparametric methods. It gets SOTA results through a dynamic architecture that adapts to changing K values and a unique loss based on new amortized inference in mixed models [79]. Our system uses DeepDPM as the primary clustering model to group the fish images.

In the case of unsupervised fish classification task, we have a set of images (input) without any labels (output) indicating the type of fish in the image. Instead of manually selecting features of the fish from the images or relying on labelled data, unsupervised learning can be applied to find patterns within the images themselves.

This can be achieved by gathering a large dataset of unlabelled images and using unsupervised learning techniques, such as clustering or dimensionality reduction, to identify groups or structures in the data.

The model learns to group the images (input) based on their similarities or inherent patterns, allowing for the classification of fish into families or species even without explicit labels. Once the model has identified these groups, new, unseen fish images can be assigned to the existing groups based on their resemblance to the patterns discovered during the learning process. This approach simplifies the classification by automating the identification process while leveraging the information contained within the unlabelled data itself.

There is limited study on fish image clustering. In our review of the research on fish image classification, we found that the majority of studies focused on supervised algorithms. Notably, Rodrigues [78] worked on the evaluation of fish clusters using five automatic fish species classification schemes based on image analysis, combining various feature extraction techniques, data clustering algorithms, and input classifiers, while Hong Yao [98] developed a fish image segmentation method that combines an improved k-means clustering algorithm with mathematical morphology to enhance accuracy and stability. Sun, Y [88] worked on an adaptive fast clustering algorithm for fish swarm image segmentation that combines the extraction of fish swarm hypoxia image features and the K-Means++ algorithm. Ibrahim [51] developed a fish image segmentation model using the Salp Swarm Algorithm (SSA) and Simple Linear Iterative Clustering (SLIC) method, with initial parameters optimized by SSA. The model works well in various situations and is better than previously used methods for finding fish in real-world images. Saifullah's [80] study uses K-means clustering for fish object detection based on color and grayscale images. In order to get fish contours, the process includes image preprocessing, segmentation with K-means clustering, and morphological processing. However, these studies primarily focus on parametric clustering techniques. According to our knowledge, this is the first study to employ non-parametric, unsupervised deep clustering of fish images and classify them into different species.

Chapter 3

Materials and Methods

In this study, only five species from the Fish4Knowledge (fish4k) dataset [4] were used, and the DeepDPM [79] clustering algorithm was applied to them. Even though there are 23 species in total in this dataset, the dataset is very unbalanced and only 10 of them have at least 200 images. In addition, the species with the most and second most images have a similar appearance so we excluded the most common species to provide a better comparison. The images were divided into train and validation sets as detailed below. Real-ESRGAN deep learning [93] approach was used to enhance the low-quality images before clustering to increase the clustering accuracy. We compared the impact of various embedding methods, such as MoCo and ViT-MAE, to assess their effectiveness. In addition, a tool was made to help see and understand the clusters better.

3.1 Datasets

As was stated in the initial research analysis, the outcomes of DeepDPM were reproduced using the MNIST [27], Fashion-MNIST [95], USPS [50], STL10 [21], and ImageNet [26] datasets in both balanced and unbalanced configurations.

Datasets Used	**Train samples**	**Val samples**	**Data dimension**	**GTK**
MNIST [27]	60,000	10,000	28×28	10
USPS [50]	7,291	2,007	16×16	10
Fashion-MNIST [95]	60,000	10,000	28×28	10
STL10 [21]	5,000	8,000	96×96×3	10
ImageNet-50 [26]	64,274	2,500	224×224×3	50

Table 3.1: Summary of the datasets used in the initial analysis

The Fish4Knowledge dataset [4] consists of 27,370 verified fish images acquired from live video footage. The dataset is divided into 23 classes, with each class represented by a species based on distinctive characteristics such as the presence or absence of components, specific number, or particular shape. The data is very imbalanced, with the most frequent species being approximately 1000 times more common than the least common. The fish images are obtained using fish detection and tracking software, and the species are manually labeled by marine biologists. Overall, the Fish4Knowledge dataset is a significant resource for study on fish recognition, tracking, and behaviour analysis, with different features representing each species and the dataset grouped into digestible clusters.

In the Fish4Knowledge dataset, only six species have at least 2,500 images each, while the remaining species have a maximum of 500 images each [4]. Although Ronen et al [79]. claimed that DeepDPM should theoretically be capable of identifying clusters of rare species, we opted to include just five of the 23 species due to the extreme imbalance and the prevalence of low-resolution images. The selected five species represent a diverse mix among the 23 species, as three of the six species with a substantial count exhibit strong similarities. The representative images of these five species are illustrated in Figure 3.1.

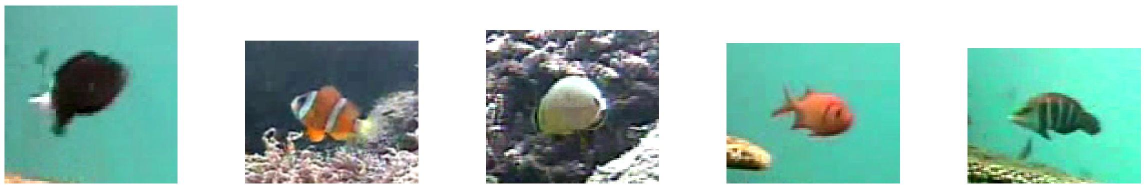
3.1.1 Train and Test Sets

The dataset was acquired from the official fish4Knowledge website, and images in each of the five folders were randomly divided into train and validation folders with an 8:2 split ratio between the train and validation sets. The order of the images within these sets was then randomised before performing the clustering to reduce training bias. The count of each species included in the train and validation sets is shown in Table 2. The species distributions in this dataset are extremely uneven, ranging from 241 total photos connected with Hemigymnus fasciatus to 4049 images associated with Amphiprion clarkii, while the validation set exhibited a similar distribution.

The images include the fish and small amount of background and are only a small portion of the original camera images that were used to identify these fish. These images are usually blurry and of low resolution. Using the ESRGAN deep learning approach, enhanced higher resolution versions of the photos were made from the train and validation folders, and this set of folders was then utilised for clustering evaluation.

The enhanced pictures were put to use in the pretraining process utilising the

MoCo and ViT-MAE algorithms to generate embeddings, which were then used to execute the unsupervised clustering approach with the DeepDPM.



Chromis chrysura　Amphiprion clarkii　Chaetodon lunulatus　Myripristis kuntee　Hemihym-nus fasciatus

Figure 3.1: Images of the five fish species used in the study

Fish Species	Train samples	Val samples
Chromis chrysura	2826	707
Amphiprion clarkii	3208	803
Chaetodon lunulatus	2045	512
Myripristis kuntee	277	70
Hemigymnus fasciatus	192	49

Table 3.2: Summary of five species of fish used in the study

3.2 Machine learning models

3.2.1 DeepDPM

DeepDPM [79] is a nonparametric deep clustering approach that, unlike the majority of deep unsupervised clustering algorithms, does not need the number of clusters to be specified beforehand. It dynamically infers and changes the number of clusters during training using split and merge method. It consists of two main components: the first is a clustering net, which produces soft cluster assignments for each input data point, while the second consists of K subclustering nets (one for each cluster k, $k \in 1, \ldots, K$). These subclustering nets generate soft subcluster assignments based on the previously generated soft cluster assignments from the clustering net [79]. Soft cluster assignments involve a probabilistic approach to assigning data points to clusters. Unlike hard clustering, where each data point is assigned to a single cluster, soft clustering assigns a degree of membership to each data point across multiple clusters. This means that a data point can belong to more than one cluster with varying degrees of membership [2] [1]. The subcluster assignments support split and merge decisions that enable the number of clusters to change on the fly. Introducing a new loss function in DeepDPM, inspired by the expectation-maximization algorithm used in Bayesian Gaussian mixture models (EM-GMM) made it more stable and efficient.

Differentiation is the computation of a function's derivative, which measures the sensitivity of a function's output to changes in its input. In machine learning, differentiation is frequently used to calculate gradients, which are subsequently used to update model parameters during training [10]. DeepDPM [79] is a deep clustering model leveraging neural networks to learn flexible data clustering. Unlike offline clustering approaches like K-means, DeepDPM is differentiable for most of the training process, enabling the computation and propagation of gradients throughout the model for gradient-based optimization. The only exception to DeepDPM's differentiability occurs when discrete splits or merges transpire within the model, as these non-differentiable operations prevent gradient computation at these points. DeepDPM's use of differentiation enables optimization using gradient-based approaches, which can help the model discover more accurate data clusterings. This might be one of the reasons why DeepDPM outperforms other clustering techniques including the traditional and deep non-parametric clustering algorithms on a wide range of datasets and metrics [79].

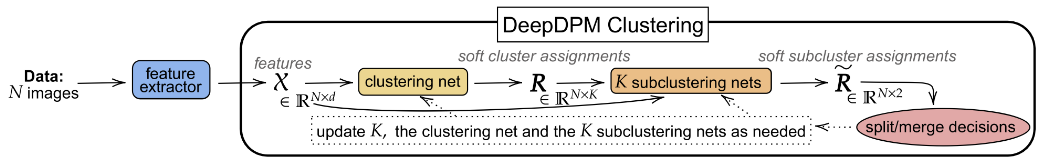


Figure 3.2: DeepDPM architecture : DeepDPM's pipeline - given features X, the clustering net outputs cluster assignments, R, while the subclustering nets generate subcluster assignments, R. Upon the acceptance of split/merge proposals, all those nets are updated during the learning. [79]

The clustering portion of the technique employs split-merge as inspired by [15] to modify the K value for clusters having a subcluster pair associated with it. There are two main components to it: the first is a clustering net made up of an MLP architecture with one input layer, one hidden layer, and an output layer (K neurons). K subclustering nets are present in the second (one for each cluster k). As in citechang2013parallel, split-merge is utilised to modify the K value during training. The model architecture, including the last layer of the clustering net, evolves as K varies. The Bayesian GMM's EM (Expectation Maximization) employs a new loss in this situation. The new amortised EM has improved point prediction across all batches, not just the current batch. As a result of the function's smoothness, locations in the observation space that are close to one another should have labels that are comparable.

Authors of DeepDPM found that the model dominated consistently across all datasets and measures, and its performance advantage only grew in unbalanced scenarios. Altogether, the results show that DeepDPM does better than other parametric, nonparametric clustering algorithms on most datasets and metrics (both classical and deep) [79]. Therefore, we chose DeepDPM as our unsupervised model for this thesis.

3.2.2 MoCo

Momentum Contrast (MoCo) is a self-supervised technique for unsupervised representation learning based on deep learning. [45]. MoCo uses contrastive learning to acquire visual representations that may be used for subsequent tasks such as image categorization. In contrastive learning, the model is taught to tell the difference between two pictures that are the same and two pictures that are different. MoCo uses momentum contrast, in particular, to do this.

Momentum contrast is a method that uses a momentum encoder to help the model learn better ways to represent things. The model maintains two encoders throughout training: a current encoder and a momentum encoder. The momentum encoder is updated by gradually averaging the current encoder's weights over time, enabling the model to acquire more stable picture representations. It has also been helpful for tasks like object detection and semantic segmentation [45] .

MoCo v2 [17], an upgraded version of the Momentum Contrast (MoCo) [45] self-supervised learning algorithm, incorporates several improvements over the original MoCo method. The improved version has a larger batch size, a better plan for negative sampling, and a dynamic queue for storing negative examples. The old 1-layer fully connected layer has been replaced with a 2-layer MLP head, and blur

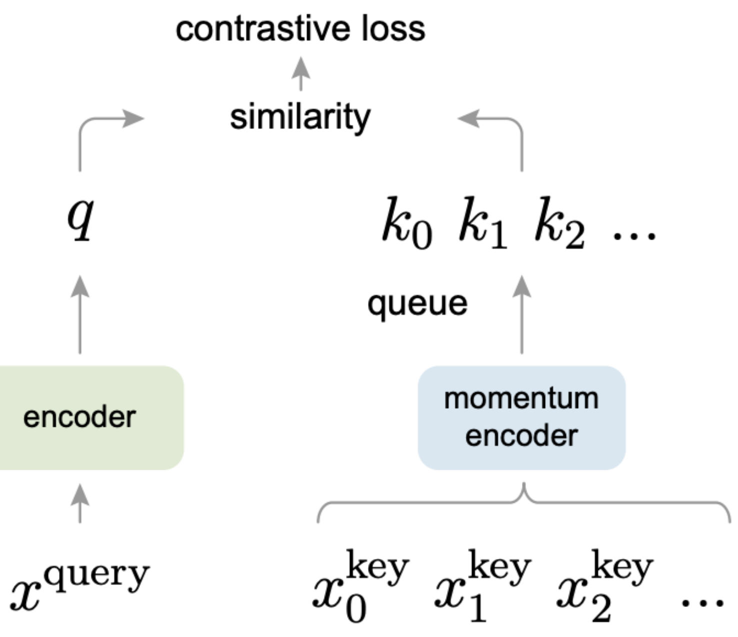


Figure 3.3: MoCo architecture - MoCo model trains a visual representation encoder by matching an encoded query q to a dictionary of encoded keys using a contrastive loss. The dictionary keys $k0$,$k1$,$k2$,... are defined on-the-fly by a set of data samples. The dictionary is built as a queue, with the current mini-batch enqueued and the oldest mini-batch dequeued, decoupling it from the mini-batch size. The keys are encoded by a slowly progressing encoder, driven by a momentum update with the query encoder. This method enables a large and consistent dictionary for learning visual representations [45]

.

enhancement has been added. These changes have made it easier to set up more reliable baselines and do better than SimCLR without needing long training batches. The training process has become more efficient and stable, resulting in improved performance on subsequent tasks. MoCo v2 [17] has achieved impressive results on several benchmarks, such as ImageNet classification, object detection, and instance segmentation. It has also been used as a pre-training method for transfer learning in many fields, such as natural language processing and medical imaging [17].

3.2.3 ViT-MAE

Masked autoencoder for vision transformer (ViT-MAE) [46] is a new methodology for training a vision transformer [30] to recognize an incomplete image and predict its original version. Vision Transformers (ViT), are a sort of deep neural network that apply the self-attention mechanism of Transformers to computer vision tasks [30]. Transformers [91] were initially developed to do problems related to natural language processing. In ViT-MAE, a set of masks is applied to the image before feeding it into the encoder transformer, which only processes the visible part of the patches. The model then learns to reconstruct the original image from the masked version through an asymmetrical encoder-decoder architecture using mask tokens and positional encodings. It uses the mean squared error (MSE) to evaluate the loss between the reconstructed and original images. The model's primary purpose is to make predictions about the raw pixel values for the areas that have been masked. Being a self-supervised pre-training model which uses autoencoders, there is no labeling requirement because the model can internally mask patches and learn how to reassemble them.

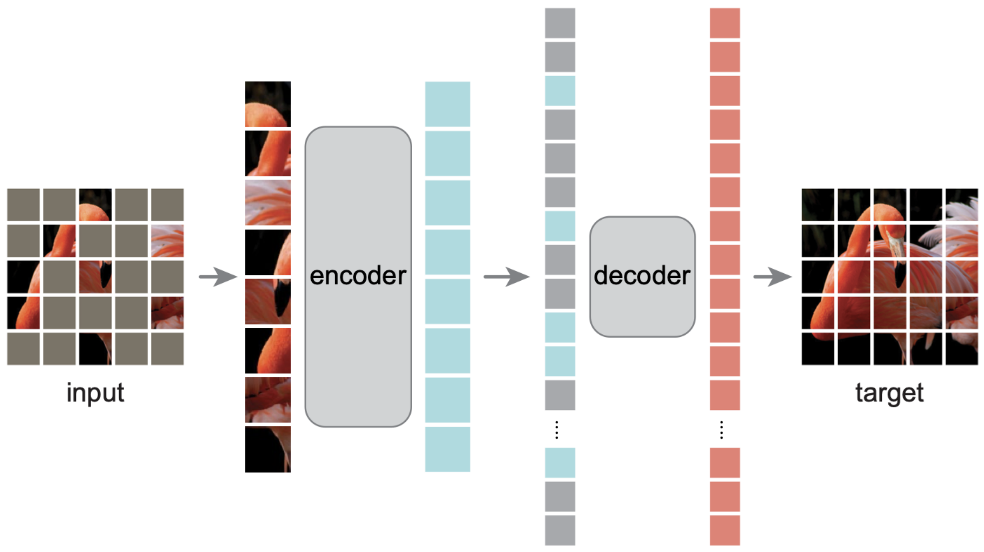


Figure 3.4: ViT-MAE architecture [46]

Coming to the main architecture of the model, the masked autoencoder employs the typical ViT (Vision Transformer) design, which consists of a stack of Transformer blocks, each of which has a multi-head self-attention block and an MLP block with LayerNorm (LN). The end of the encoder is LN. After the encoder, a linear projection layer is utilised to accommodate the varying widths of the MAE (Multi-Adversarial

Encoder) encoder and decoder. The MAE adds sine-cosine positional embeddings to both the encoder and decoder inputs. The network does not employ layer scaling or relative positioning. To accommodate ViT's class token, an auxiliary dummy token is added to the encoder input during MAE pre-training; this token will be used as the class token for training the classifier in linear probing and fine-tuning. The MAE functions properly with or without the token.

The performance of the MAE (masked autoencoder) model was compared with other techniques, such as DINO, MoCov3, or BEiT, after pre-training on ImageNet-1K and fine-tuning it end-to-end.

ViT-MAE embeddings are obtained by training a ViT model with a masked autoencoder (MAE) objective. The MAE objective is to mask random patches of the input image and reconstruct the missing pixels. This forces the model to learn rich visual representations that can capture both local and global information. MoCo embeddings are obtained by training a convolutional neural network model with a contrastive learning objective. The contrastive learning objective is to distinguish between positive and negative pairs of image patches based on their similarity in a latent space. This encourages the model to learn invariant features that can generalize across different views or augmentations.

Some benefits of employing ViT-MAE embeddings as opposed to MoCo embeddings [45] include:

- ViT-MAE embeddings do not require large batch sizes or memory banks for contrastive learning, which makes them more efficient and easier to train.
- ViT-MAE embeddings are not dependent on data augmentations or hard negative mining, which can add biases or noise into the training process.

3.2.4 Real-ESRGAN

Most of fish images present are small and blurry which resulted in poor classification performance. To deal with this, Real-ESRGAN [93] was used as an image enhancer to increase the resolution of the fish images with the goal of improving classification performance. Real-ESRGAN is a state-of-the-art image super-resolution algorithm based on an enhanced version of the ESRGAN architecture. Enhanced Super-Resolution

Generative Adversarial Network (ESRGAN) [94] is a deep learning-based method used for increasing the resolution of low-quality images. It works by using a generator and discriminator network both made up of deep convolutional neural networks (CNNs) [62] to improve the quality of the generated image. The generator network uses a series of residual blocks and feature fusion modules to learn high-level image features and create realistic-looking high-resolution images.

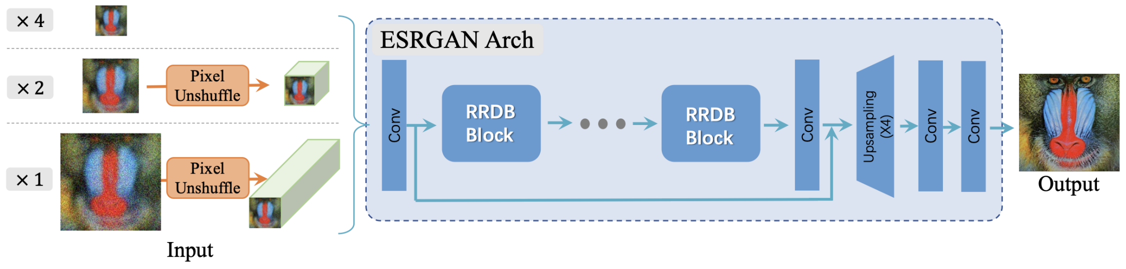


Figure 3.5: Real-ESRGAN architecture [93]

ESRGAN in fact is an improved version of SRGAN [64]. ESRGAN attempted to improve SRGAN by altering its model's architecture and loss mechanisms. SRGAN, like other GANs [40], consists of two algorithms: one that generates a picture and another that determines if the image is real or fake. GANs (Generative Adversarial Networks) [40] are a type of generative, unsupervised neural networks that approximate the data-generating distribution used to make a given dataset [40]. They have a generator (G) and a discriminator (D) network [40]. The generator makes new pieces of data similar to the training data, and the discriminator decides whether a piece of data is accurate. In a zero-sum game, the two networks are trained together. The generator tries to trick the discriminator by making more realistic data, while the discriminator tries to get better at spotting fake data [40]. Comparing a generated image against a genuine image, ESRGAN determines which is more realistic.

ESRGAN can be a useful tool for improving the quality of low-quality fish images by generating high-quality, super-resolved images. These images can provide more detailed information and make it easier to accurately identify and classify different species of fish.

3.2.5 Training

The training of the DeepDPM clustering model typically requires two steps. First, embeddings are created from the images, which are then used as inputs to the main clustering model. There are several ways to create these embeddings, including using the deep unsupervised feature extractors MoCo or SimClr [16], as described in the original paper. The feature extractor can be trained from scratch for the given data or pre-trained weights can be used. If the dataset is small, the feature extractor can be skipped and UMAP can be applied directly to the raw data or an autoencoder followed by a UMAP [71] can be used to generate the low-dimensional representations of the images to be clustered. Although the DeepDPM clustering algorithm can work on higher dimensions, the authors recommend keeping the maximum dimension to 128D. If the dataset is relatively low-dimensional (< 128D), it is possible to train the algorithm on the raw data itself. Before passing the embeddings to the clustering stage, an autoencoder [48] must be trained on top of the embeddings. An autoencoder is a type of artificial neural network used for unsupervised learning. It learns to produce a compressed representation of input data by training itself to ignore signal 'noise' . It has three layers: an input layer, a hidden layer for encoding, and an output decoding layer [48]. If the datasets mentioned in the original DeepDPM paper are used, they already have pre-trained autoencoder weights. Thus, for any new datasets we perform end-to-end training in the script, training an autoencoder on top of the MoCo embeddings, including the feature extraction pipeline where learning of clustering and features takes place. For implementing DeepDPM the original implemenation has been used which can be found on the GitHub [13].

For our use case, we compared embeddings generated using MoCo from scratch against pre-trained MoCo weights that were trained on the ImageNet dataset. In all experiments, the clustering net is configured with 50 hidden units. A batch size of 128 is used, and the clustering net is trained with a learning rate of 0.002, while the subclustering nets use a learning rate of 0.005. The end-to-end DeepDPM alternation clustering model was trained for 400 epochs. The total training time for this model was approximately. For performing the MoCo pretraining from scratch lightly's implementation has been used which can be found on the documentation page [68].

The original authors of ViT-MAE found that masking a high proportion of the input image, such as 75, yields a nontrivial and meaningful self-supervisory task. The target for training contained normalized pixel values. The total training batch size was calculated as the product of training-args-train-batch-size, training-args-gradient-accumulation-steps, and training-args-world-size. The base-learning-rate was set to 1.5e-4, and the model was trained with 100 epochs. For training ViT-MAE embeddings Hugging face's image-pretraining has been used, which can be found on its GitHub [34] In addition to this Real-ESRGAN was used to improve the quality of images of fish4knowledge. For this, the original github documentation of Real-ESRGAN was used. Default scale factor of 3.5 was used to enhance the images. For performing image enhancement using the Real-ESRGAN, the original source code has been used which can be found on Github [92].

3.3 Metrics

Unsupervised clustering models are difficult to evaluate since there are no goal values or labels to compare with [65]. Unsupervised models, in contrast to supervised models, lack a defined objective function or performance metric that can be improved or measured [38]. As a result, assessing unsupervised clustering models necessitates a variety of methodologies that are dependent on the data and the purpose of the analysis [54]. As a result, we decided to use a dataset consisting of fish in which the target value is already present and evaluated it using supervised clustering metrics.

3.3.1 Evaluating DeepDPM

For evaluation metrics, we employed the same clustering metrics as DeepDPM [79]. These include the three main ways of measuring the efficacy of supervised clustering evaluations namely Accuracy, NMI and ARI. Here, the accurary measures the proportions of instances that are correctly clustered, while the NMI and ARI tell us about the quality of the clusters. A point that should be considered while evaluating the clustering model's performance is that there may be certain instances, particularly with highly imbalanced datasets, where all images may be classified into one or two clusters. In such situations, the resulting clusters may be meaningless, even if the accuracy is high. So, while looking at the overall clustering performance, it is better to take a holistic approach where we consider multiple factors, including the ARI and NMI values. Suppose if these values are close to or near zero, they indicate that the clusters are either randomly assigned or of poor quality.

Clustering Accuracy (ACC), is represented as

$$\text{ACC} = \max_{m} \frac{\sum_{i=1}^{N} 1(y_i = m(z_i))}{N} \tag{3.1}$$

where N is the total number of data points and y_i represents the Ground-Truth (GT) class label associated with each data point. i, z_i represents the forecasted cluster assignment in accordance with the clustering algorithm that is being considered, $\mathbb{1}(\cdot)$ is the indicator function, and m is defined by all of the feasible one-to-one mappings that exist between the anticipated class membership and the actual one. DeepDPM uses the Hungarian algorithm to determine how accurate a cluster is by fixing problems with how the predicted labels match the actual labels. It generates a confusion matrix, identifies the optimal one-to-one mapping between labels to minimize discrepancies, and computes the accuracy by dividing correct assignments by the total number of data points. It effectively represents clustering performance while accounting for potential label assignment discrepancies.;

Normalized Mutual Information (NMI) is defined by

$$\text{NMI} = \frac{2 \times I(y; z)}{H(y) + H(z)} \tag{3.2}$$

where $H(.)$ represents entropy and $I(.;.)$ stands for mutual information (MI). The fact that this measure does not take into account high cardinalities while calculating the MI term, which is included in the numerator, is one of its shortcomings (i.e., over clustering). The NMI does not have a sensitive enough detection system for overclustering.;

Adjusted Rand Index (ARI) : The Rand index (RI) is a metric that is used to quantify the percentage of "right" judgements made for each pair of data points. If two instances come from the same GT class and are given the same cluster assignment (also known as a true positive, or TP), then a choice was made correctly. However, if the examples came from separate GT classes and were given different clusters, then the decision was incorrect (a true negative, TN). In a similar manner, clustering mistakes may be divided into two categories: false positives (FP) and false negatives (FN) (FN). Hence, RI may be calculated as follows:

$$\mathrm{RI} = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{TN} + \mathrm{FP} + \mathrm{FN}} \tag{3.3}$$

The Rand index provides the basis for the ARI measure, which is an adjusted version of the original index. Given a set $\mathcal{S}$ of N elements, and two groupings or partitions (e.g. y and z) of these elements, a contingency table can show the overlap between y and z. table $[c_{kl}]$ where each value c_{kl} is the number of things that are shared by both y_k and z_k : $c_{kl} = |y_k \cap z_k|$. Let a_k be the sum (f each row, meaning, $a_k = \sum_l c_{kl}$, and b_k the sum of each column, i.e. $b_k = \sum_k c_{kl}$.

The ARI measure is then computed by:

$$\mathrm{ARI} = \frac{\sum_{k,l}\binom{n_{kl}}{2} - \left[\sum_k \binom{a_k}{2} \sum_l \binom{b_l}{2}\right] / \binom{n}{2}}{\frac{1}{2}\left[\sum_k \binom{a_k}{2} + \sum_l \binom{b_l}{2}\right] - \left[\sum_k \binom{a_k}{2} \sum_l \binom{b_l}{2}\right] / \binom{n}{2}} \tag{3.4}$$

The greater their values, the better.

3.3.2 Evaluating ESRGAN

The Real-ESRGAN model uses a combination of four loss functions during training to generate high-quality super-resolved images. These loss functions include: L1 loss: The L1 loss measures the absolute difference between the generated image and the ground truth image on a pixel-by-pixel basis. The formula for L1 loss is:

$$L1(x, y) = ||x - y||_1 \tag{3.5}$$

where x and y are the generated image and the ground truth image, respectively; and $||.||_1$ denotes the L1 norm. Perceptual loss: The perceptual loss measures the difference between the high-level features of the generated image and the ground truth image. It is calculated using a pre-trained deep neural network that extracts feature maps from the input images. The formula for perceptual loss is:

$$Perceptual(x, y) = ||\Phi(x) - \Phi(y)||_1 \tag{3.6}$$

where $\Phi(.)$ denotes the feature extraction function; and $||.||_1$ denotes the L1 norm. Adversarial loss: The adversarial loss measures how well the generated image can fool the discriminator network, which is trained to distinguish between real and generated images. The formula for adversarial loss is:

$$Adversarial(x) = -\log(D(G(x))) \tag{3.7}$$

where x is the generated image; $G(.)$ denotes the generator function; $D(.)$ denotes the discriminator function; and $\log(.)$ denotes the natural logarithm. Total Variation (TV) loss: The TV loss measures the total variation of the generated image, which is used to preserve the edges and fine details of the original image. The formula for TV loss is:

$$TV(x) = ||\nabla x||_1 \tag{3.8}$$

where ∇x denotes the gradient of the generated image with respect to its spatial coordinates; and $||.||_1$ denotes the L1 norm. The overall loss function for the Real-ESRGAN model is a weighted sum of these individual losses, as follows:

$$Totalloss(x, y) = w_1 * L1(x, y) + w_2 * Perceptual(x, y) + w_3 * Adversarial(x) + w_4 * TV(x) \tag{3.9}$$

where w_1, w_2, w_3, and w_4 are the weights assigned to each loss term, respectively.

3.3.3 Evaluating ViT-MAE

The Mean Squared Error (MSE) is a loss function used in the training of masked autoencoder for vision transformer to measure the difference between the reconstructed output and original input images in the pixel space. Similar to BERT, the loss is computed exclusively on masked patches. The formula for MSE loss is:

$$MSE(y_{true}, y_{pred}) = \frac{1}{N} \sum (y_{true} - y_{pred})^2 \tag{3.10}$$

where y_{true} is the ground truth input image, y_{pred} is the output image generated by the autoencoder, and N is the total number of pixels in the image. The objective is to reduce the MSE loss during training, which is achieved by modifying the autoencoder's weights through backpropagation. The autoencoder may learn to create output pictures that are as comparable as feasible to the input images by minimizing the MSE loss.

Chapter 4

Results

All the experiments were conducted on Google Colab Pro having a single Tesla T4 gpu. The training process, which included creating embeddings from the MoCo pre-trained model and performing clustering, required approximately 6 hours. In contrast, training the MoCo model from scratch and then clustering consumed around 10 hours. The time taken for ViT-MAE was comparable to that of training MoCo from scratch.

4.1 Initial results

Table 4.1 displays the findings that were replicated on both balanced and imbalanced MNIST [27], Fashion-NIST [95], USPS [50], STL10 [21], and ImageNet [25] datasets.

As per the original paper [79], for the MNIST, USPS, and Fashion-MNIST datasets, as well as their unbalanced versions, the same (and fixed) data embeddings were used as input, and parametric clustering was done with ground truth K given to them and compared with DeepDPM. It can be observed from the results in Table 4.1 that the algorithm was usually successful in estimating the value of K for each of the datasets. Surprisingly, we detected fewer clusters and with reduced accuracy than reported by the authors for both USPS datasets. In addition, as reported by the authors, we observed that DeepDPM may overestimate the number of clusters on the ImageNet dataset. The model underestimated the K value for the USPS dataset, but the K value was overstated for the ImageNet dataset. The original value of K for the ImageNet dataset was 50, but the value of K in the paper and in the new results was 52. In the USPS dataset, the original value of K was 10, but the value of K in the paper and the new results were 9 and 7. Similarly, for the USPS-imbalanced dataset, K value in the paper and the new results were 9 and 8. Thus, it became apparent that replicating the results of DeepDPM on the USPS dataset proved to be challenging. However, by training the model for an extended duration and increasing the number of epochs, there is potential for the model to converge towards the accurate value of value of K. The Acc column indicates the accuracy of the clustering findings, whereas the PaperAcc column indicates the accuracy stated in the original study DeepDPM research paper. In general, the Acc scores were marginally lower than the PaperAcc scores, showing a certain degree of variation in the clustering results. Only, the USPS, USPS-imbalanced datasets showed a noticeable gap between Acc and PaperAcc. In the USPS dataset, the Acc was 0.727, while the Paper Acc was 0.89, indicating a difference of 0.163. Similarly, in the USPS-imbalanced dataset, the Acc was 0.809, while the Paper Acc was 0.94, indicating a difference of 0.131. The performance of DeepDPM was affected by the imbalance of the datasets, with better performance achieved on balanced datasets. Overall, the difference between the original K values and the values used in the paper and in the new results is small, ranging from 1 to 2.

Despite observing some differences, as detailed above, DeepDPM's initial results were quite promising, as the model was able to figure out the exact number of clusters most of the time without being told. So far, DeepDPM has only been tested on standard datasets and not on real datasets of fish. The next step was to test the algorithm on a set of real fish in fish4knowledge.

Table 4.1: Results obtained by evaluating the DeepDPM algorithm on different datasets.

Datasets Used	Evaluation Metrics						
	NMI	ARI	Acc	PaperAcc	K	PaperK	OrgK
MNIST	0.941	0.953	0.9787	0.98	10	10	10
MNIST-imbalanced	0.941	0.953	0.978	0.98	10	10	10
Fashion-MNIST	0.687	0.534	0.663	0.62	10	10	10
Fashion-MNIST-imb	0.670	0.515	0.612	0.61	10	10	10
USPS	0.826	0.70137	0.727	0.89	7	9	10
USPS-imbalanced	0.864	0.807	0.809	0.94	8	9	10
STL 10	0.768	0.670	0.832	0.85	10	10	10
ImageNet	0.736	0.515	0.645	0.66	52	52	50

4.2 Image Enhancement

Our initial findings from clustering the original fish4knowledge dataset using DeepDPM were not impressive. We hypothesised that, given that DeepDPM operates on extracted picture features, the majority of image pixel values would fall within a 50 to 250 range. These blurry low resolution underwater camera images may be of insufficient quality to extract distinguishing characteristics from, preventing the clustering algorithm from classifying them properly based on species. So, we chose to apply an image enhancement technique to the original fish4knowledge dataset's photos. After conducting some research, we determined that the Real-ESRGAN model can serve our needs.

Real-ESRGAN is an effective way to increase the resolution of low-quality fish images, resulting in improved image quality. This can help intensify the features of the fish, such as their scales and fins, and help better identify the fish species. Moreover, ESRGAN is a cost-effective and time-efficient method to enhance the quality of low-quality fish images. It can be applied directly to poorly captured images, thus reducing the need for expensive equipment and saving time and effort in capturing new images, mainly when dealing with a large volume of images.

Pixel intensity refers to the value associated with each pixel in an image [5] [6]. Following picture enhancement, the range of pixel intensities grew from 50-250 to 500-2500. This may be seen by comparing the before and after photographs in figures 4.3 and 4.4. The average size of images also increased from being in the range of 50-150 before enhancement figure 4.1 to 200-500 after enhancement figure 4.2.

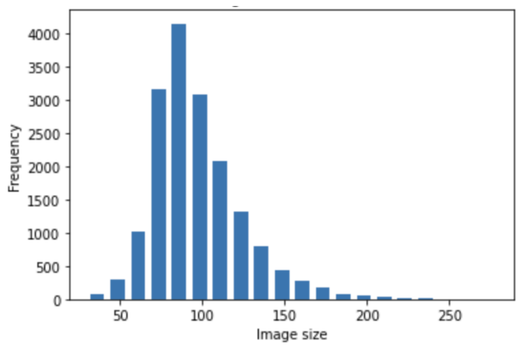


Figure 4.1: Image size distribution before enhancement

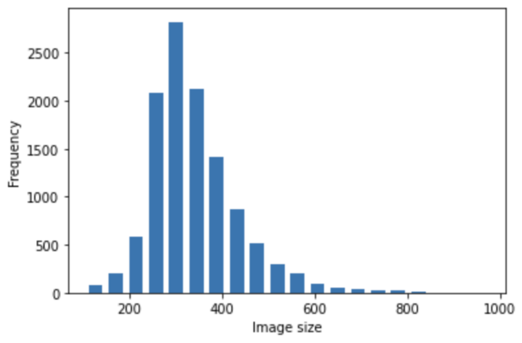


Figure 4.2: Image size distribution after enhancement

4.3 DeepDPM results using MoCo before enhancement

Table 4.2 and Table 4.3 show the results of training and validation experiments for the DeepDPM clustering algorithm using the pre-trained MoCo embeddings trained on

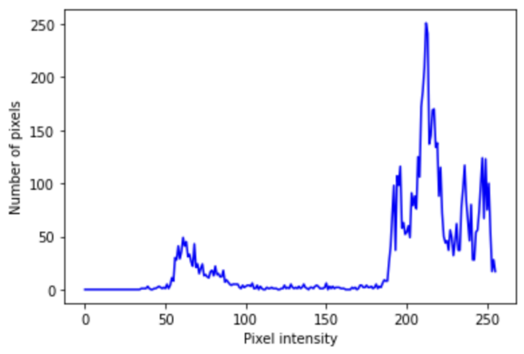


Figure 4.3: Pixel intensity distribution before enhancement

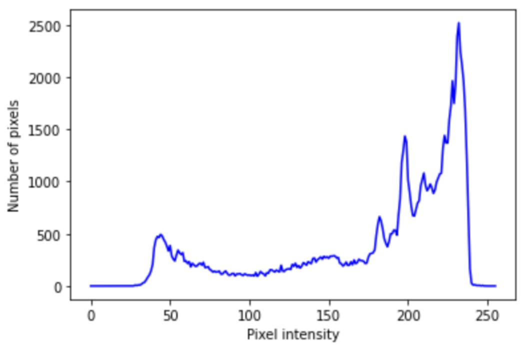


Figure 4.4: Pixel intensity distribution after enhancement

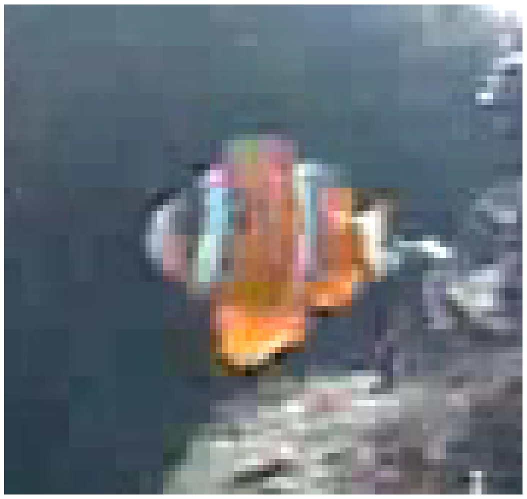


Figure 4.5: Amphiprion clarkii before enhancement using Real-ESRGAN

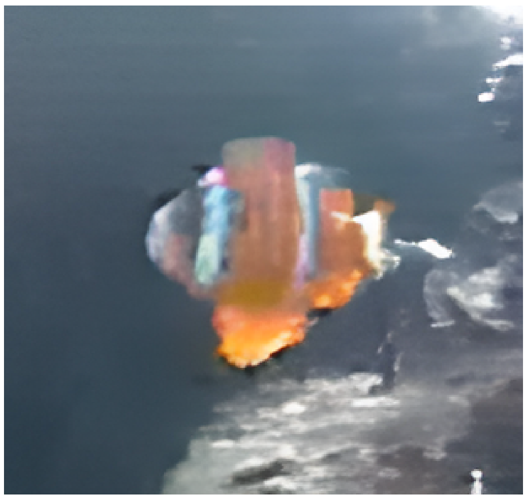


Figure 4.6: Amphiprion clarkii after enhancement using Real-ESRGAN

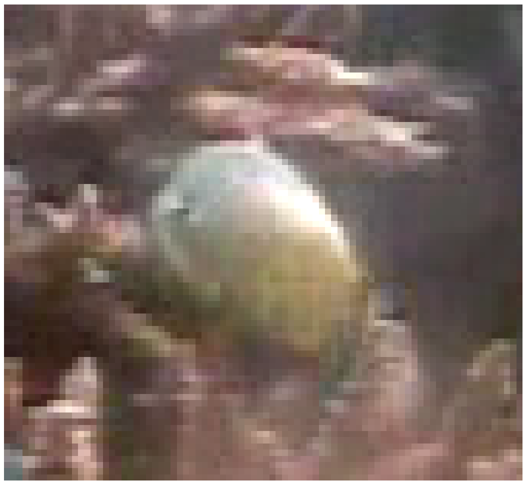


Figure 4.7: Chaetodon lunulatus before enhancement using Real-ESRGAN

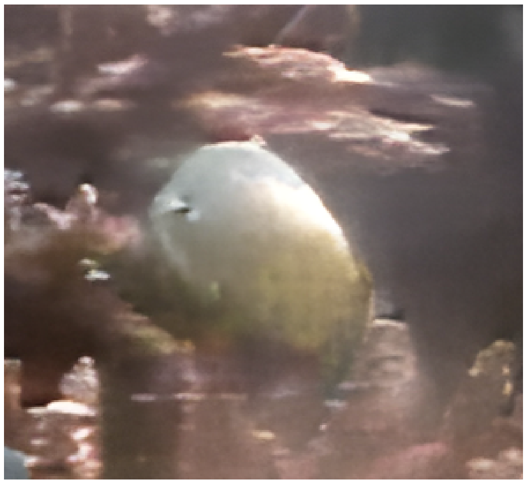


Figure 4.8: Chaetodon lunulatus after enhancement using Real-ESRGAN

Table 4.2: Results of MoCo model pretrained on ImageNet

Train							
Exp No	NMI	ARI	Acc	Final K	Org K	L-Dim	Epochs
1	0.02 ± 0.031	0.01 ± 0.032	0.38 ± 0.013	1.66 ± 0.57	5	32	400
2	0.0 ± 0	0.0 ± 0	0.37 ± 0.001	1.33 ± 0.57	5	64	400
3	0.005 ± 0.008	0.02 ± 0.035	0.38 ± 0.017	2 ± 1	5	128	400
Validation							
Exp No	NMI	ARI	Acc	Final K	Org K	L-Dim	Epochs
1	0.001 ± 0.002	0.0 ± 0.0	0.37 ± 0.001	2 ± 0	5	32	400
2	0.0 ± 0.0	0.0 ± 0.0	0.37 ± 0.0	1.33 ± 0.57	5	64	400
3	0.02 ± 0.033	0.02 ± 0.032	0.38 ± 0.014	2 ± 1	5	128	400

Imagenet and the MoCo embeddings trained on the fish image dataset from scratch, respectively. The "Experiment No" (Exp No) column implies different runs of the DeepDPM model on the same dataset. Each experiment is executed under varying settings of latent dimensions (L-dim) and epochs, factors that may impact the final result. The table presents the results of three different settings, showcasing the mean and standard deviations of three independent runs for all performance metrics, including K.

The results displayed in Table 4.2 reveal that the NMI, ARI, and accuracy values for all three experiments are relatively low, suggesting suboptimal clustering algorithm performance. The performance remained consistently low across the experiments regardless of the latent size suggesting that this factor did not impact the

Table 4.3: Results of MoCo model trained from scratch on fish4k dataset

Train							
Exp No	NMI	ARI	Acc	Final K	Org K	L-Dim	Epochs
1	0.31 ± 0.078	0.28 ± 0.085	0.56 ± 0.048	2.33 ± 0.57	5	32	400
2	0.34 ± 0.09	0.29 ± 0.099	0.57 ± 0.043	3.66 ± 0.57	5	64	400
3	0.19 ± 0.038	0.15 ± 0.044	0.50 ± 0.027	2.33 ± 0.57	5	128	400
Validation							
Exp No	NMI	ARI	Acc	Final K	Org K	L-Dim	Epochs
1	0.31 ± 0.064	0.28 ± 0.076	0.56 ± 0.043	2.33 ± 0.57	5	32	400
2	0.33 ± 0.082	0.29 ± 0.093	0.57 ± 0.041	3.66 ± 0.57	5	64	400
3	0.20 ± 0.040	0.17 ± 0.047	0.50 ± 0.028	2.33 ± 0.57	5	128	400

clustering process. The final K value for all experiments reaches up to 2, which is considerably different from the original K value of 5. In particular, the final K value is approximately 2 for experiments 2 and 3, and around 3 for experiment 3. This indicates that not only did the algorithm struggle to perform well, but it also failed to identify the correct number of clusters within the dataset. The NMI and ARI scores of the three experiments in Table 4.2 are more or less zero. Moreover, the accuracy remains relatively constant across all experiments, with a value of approximately 0.38. This behaviour is also reflected in the validation results, further supporting the observation.

In contrast, the results in Table 4.3 show an improvement in the performance of the clustering algorithm when using the embeddings trained from scratch on the dataset. In all three settings, we observed much higher NMI, ARI, and accuracy values than in the previous table. The NMI and ARI scores of the three experiments in 4.3 have reached an average value of 0.28 and 0.24. The highest NMI score observed was 0.34, which is in experiment 2. Similarly, the accuracy score improved when training from scratch, with the highest score of 0.57 achieved in experiment 1. All the experiments ended up with K values of around 3, which is still less than the original K value of 5. In particular, experiment 3 in Table 4.3 shows the best performance, with an NMI value of 0.34, an ARI value of 0.29, and an accuracy value of 0.57. These values are much higher than the corresponding values in Table 4.2. It is also observed that as the latent dimensions increase, the model performance slightly improves up to 64 dimensions. However, when the latent dimensions reach 128, the performance experiences a decline. This trend is more evident when examining the NMI and ARI values, which exhibit approximately a 45% drop in percentage. The accuracy demonstrates a similar behavior. The final K value showed only a slight variations in these experiments. Likewise, the measures in the validation data, as presented in Table 4.3, exhibit a comparable pattern.

Overall, these results show that the MoCo embeddings trained on the given fish dataset were better at grouping the fish images in the Fish4Knowledge dataset than the MoCo embeddings pre-trained on ImageNet.

4.4 DeepDPM results using MoCo after enhancement

Comparing the two tables, we can observe that the clustering performance is better in Table 4.5 than in Table 4.4, as the NMI, ARI, and accuracy values are higher in Table 4.5. The highest average NMI value in Table 4.4 is 0.046, while in Table 4.5, it is 0.35. The highest average ARI value in Table 4.4 is 0.025, while in Table 4.5, it is 0.31. The highest average accuracy value in Table 4.4 is 0.38, while in Table 4.5, it is 0.60. The NMI, ARI, and Acc values in Table 4.5 are higher than those in Table 4.4 across all experiments. Furthermore, the final number of clusters consistently approximates 3 in Table 4.5, which implies that the clustering algorithm can identify better clusters compared to Table 4.4, where the final number of clusters varies between 2 and 4. Based on these metrics, Table 4.5 outperforms Table 4.4 on both the training and validation datasets.

A similar pattern was observed in Table 4.5, where the NMI and ARI values experienced a slight increase as the latent dimensions expanded and subsequently decreased when the dimensions reached 128. However, the accuracy did not follow this trend; it continued to increase as the dimensions grew. This observation suggests that the latent dimensions indeed had an impact on the clustering performance, as previously noted before the enhancement was made.

In conclusion, Table 4.5 performs better than Table 4.4 regarding NMI, ARI, accuracy and K; it indicates that the MoCo embeddings trained from scratch are better than the MoCo models trained with ImageNet weights.

Table 4.4: Results of MoCo model pretrained on ImageNet after enhancement

Train							
Exp No	NMI	ARI	Acc	Final K	Org K	L-Dim	Epochs
1	0.046 ± 0.019	0.02 ± 0.015	0.33 ± 0.013	2 ± 0	5	32	400
2	0.025 ± 0.016	0.018 ± 0.001	0.38 ± 0.012	1.66 ± 0.57	5	64	400
3	0.035 ± 0.01	0.025 ± 0.021	0.38 ± 0.003	2 ± 1	5	128	400
Validation							
Exp No	NMI	ARI	Acc	Final K	Org K	L-Dim	Epochs
1	0.040 ± 0.007	0.019 ± 0.025	0.32 ± 0.0291	2 ± 0	5	32	400
2	0.019 ± 0.001	0.019 ± 0.002	0.38 ± 0.009	1.66 ± 0.57	5	64	400
3	0.034 ± 0.037	0.025 ± 0.021	0.37 ± 0.01	2 ± 1	5	128	400

Table 4.5: Results of MoCo model trained from scratch on fish4k dataset after enhancement

Train							
Exp No	NMI	ARI	Acc	Final K	Org K	L-Dim	Epochs
1	0.29 ± 0.024	0.26 ± 0.062	0.60 ± 0.028	2.33 ± 0.57	5	32	400
2	0.35 ± 0.093	0.31 ± 0.053	0.57 ± 0.035	3.66 ± 0.57	5	64	400
3	0.21 ± 0.082	0.15 ± 0.039	0.51 ± 0.033	3.33 ± 0.57	5	128	400
Validation							
Exp No	NMI	ARI	Acc	Final K	Org K	L-Dim	Epochs
1	0.32 ± 0.049	0.26 ± 0.009	0.60 ± 0.013	2.33 ± 0.57	5	32	400
2	0.34 ± 0.034	0.30 ± 0.097	0.57 ± 0.021	3.66 ± 0.57	5	64	400
3	0.21 ± 0.004	0.19 ± 0.078	0.51 ± 0.046	3.33 ± 0.57	5	128	400

4.5 Comparing results before and after Real-ESRGAN enhancement

Tables 4.2, 4.3, 4.4, and 4.5 illustrate the performance contrast in terms of image enhancement between a MoCo model pre-trained on ImageNet (Tables 4.2 and 4.4) and a MoCo model trained from scratch on the fish4k dataset (Tables 4.3 and 4.5).

While comparing Tables 4.2 and 4.4, we can observe that pre-training on ImageNet results in lower performance before the image enhancement. The NMI values in the training and validation experiments range between 0.0 and 0.02, the ARI between 0.0 and 0.02, and the accuracy between 0.37 and 0.38. After performing the image enhancement, we can notice the performance improvement (Table 4.4) for all three parameters. The NRI, ARI, and accuracy increased to a range between 0.025 and 0.046, 0.018 and 0.025, and 0.33 and 0.38, respectively. The number of clusters obtained on the dataset became consistent after enhancement.

Similarly, comparing tables 4.3 and 4.5, we can observe that the NMI, ARI, and accuracy for the training and validation datasets before enhancement ranged between

0.19 and 0.34, 0.15 and 0.29, and 0.50 and 0.57, respectively, resulting in 3 clusters in all the cases. In contrast, the results after enhancement show better performance. After enhancement, the NMI values range from 0.21 to 0.35, the ARI values range from 0.15 to 0.31, and the accuracy values range from 0.51 to 0.60, with the final number of clusters reaching up to 4 in some cases (Table 4.5).

When comparing the K values to earlier experiments without enhancement, they are slightly higher, indicating that both MoCo pretraining from scratch and image enhancement have an influence on the choice of K.

Overall upon comparing the results from these tables to those from previous tables before enhancement, it is seen that Real-ESRGAN image enhancement has a positive impact on clustering performance. And the MoCo model trained from scratch on the fish4k dataset results in better clustering performance than pre-training on a larger, more general dataset such as ImageNet.

4.6 Use of ViT-MAE embeddings

Despite the picture enhancements, the clustering model still did not perform well enough to be used to automatically cluster fish images for labeling. To address this, we decided to try a new method of picture embedding extraction. After attempting a few various embeddings and realising that MoCo, as utilised by the developers of DeepDPM, was not well suited to low resolution fish images, we settled on utilising the embeddings acquired using ViT-MAE.

4.7 DeepDPM results before and after enhancement with ViT-MAE

The below tables, Table 4.6 and Table 4.7 present the results of DeepDPM clustering experiments conducted with ViT-MAE embeddings using different settings before and after applying the Real-ESRGAN image enhancement technique.

The experiments were performed on the same five classes of the Fish4Knowledge dataset for consistency, and the models were evaluated based on NMI, ARI, accuracy, and K value. Although Experiment 2, which used a latent dimension of 64 for 400 epochs, had the best overall performance in terms of NMI, ARI, and K values (0.64, 0.63, and 4 in Table 4.6, and 0.75, 0.74, and 4 in Table 4.7), Table 4.7 generally had higher NMI and ARI values for other experiments. Additionally, Table 4.7 had higher accuracy values for most experiments, with the highest accuracy being in Experiment 3 at 0.84 and 0.84 in Experiment 5 in Tables 4.6 and 4.7, respectively. The accuracy values showed a slight increase in most of the cases except in experiments 2 and 3. Regarding the final K value, Table 4.6 and Table 4.7 have a similar range of values (3-5) for most of the experiments, and there is little difference in the average value of the final K between the two tables. However, the number of matches between the final K and the original K value, which is the number of clusters in the original dataset, is higher in Table 4.7 than in Table 4.6 for all experiments, indicating that Table 4.7 can match the original K value, thus producing better clustering results than Table 4.6.

Table 4.6: Results obtained using ViT-MAE embeddings before image enhancement under different settings

Train							
Exp No	NMI	ARI	Acc	Final K	Org K	L-Dim	Epochs
1	0.47 ± 0.023	0.42 ± 0.003	0.69 ± 0.006	4 ± 0	5	64	200
2	0.64 ± 0.069	0.63 ± 0.089	0.79 ± 0.007	4.33 ± 1.15	5	64	400
3	0.63 ± 0.009	0.61 ± 0.028	0.84 ± 0.001	3 ± 0	5	64	800
4	0.59 ± 0.052	0.57 ± 0.062	0.78 ± 0.025	4.6 ± 1.15	5	32	400
5	0.60 ± 0.065	0.55 ± 0.050	0.76 ± 0.065	3.66 ± 0.57	5	128	400
Validation							
Exp No	NMI	ARI	Acc	Final K	Org K	L-Dim	Epochs
1	0.45 ± 0.048	0.41 ± 0.016	0.68 ± 0.011	4 ± 0	5	64	200
2	0.63 ± 0.068	0.62 ± 0.089	0.80 ± 0.004	4.33 ± 1.15	5	64	400
3	0.62 ± 0.001	0.61 ± 0.002	0.83 ± 0.003	3 ± 0	5	64	800
4	0.58 ± 0.048	0.56 ± 0.063	0.78 ± 0.027	4.6 ± 1.15	5	32	400
5	0.59 ± 0.052	0.54 ± 0.044	0.76 ± 0.063	3.66 ± 0.57	5	128	400

The results shows that the performance in Table 4.7 is generally better than that in Table 4.6 in terms of NMI, ARI majorly and also accuracy. The higher original K values in Table 4.7 also suggest that it produces more precise clustering results than Table 4.6. This means that the models in Table 4.7 were better at classifying the fish images into their respective species.

Experiments 1, 2, and 3 have the same latent dimension (64) but different epoch counts (200, 400, and 800, respectively). The clustering performance (NMI, ARI, and Acc) typically improves as the number of epochs grows, but Experiment 2 resulted in the most outstanding performance in both tables. It achieved the highest performance

Table 4.7: Results obtained using ViT-MAE embeddings after image enhancement under different settings

Train							
Exp No	NMI	ARI	Acc	Final K	Org K	L-Dim	Epochs
1	0.64 ± 0.085	0.62 ± 0.117	0.73 ± 0.087	6 ± 1	5	64	200
2	0.75 ± 0.011	0.74 ± 0.006	0.83 ± 0.006	4.66 ± 0.57	5	64	400
3	0.70 ± 0.005	0.68 ± 0.012	0.79 ± 0.009	4 ± 0	5	64	800
4	0.72 ± 0.027	0.72 ± 0.030	0.82 ± 0.019	4 ± 1	5	32	400
5	0.67 ± 0.004	0.61 ± 0.005	0.84 ± 0.002	3 ± 0	5	128	400
Validation							
Exp No	NMI	ARI	Acc	Final K	Org K	L-Dim	Epochs
1	0.64 ± 0.084	0.60 ± 0.154	0.69 ± 0.081	6 ± 1	5	64	200
2	0.74 ± 0.009	0.74 ± 0.006	0.83 ± 0.005	4.66 ± 0.57	5	64	400
3	0.69 ± 0.002	0.68 ± 0.004	0.79 ± 0.008	4 ± 0	5	64	800
4	0.69 ± 0.011	0.71 ± 0.029	0.81 ± 0.024	4 ± 1	5	32	400
5	0.66 ± 0.002	0.60 ± 0.003	0.84 ± 0.001	3 ± 0	5	128	400

metrics (NMI, ARI), and the accuracy is almost close to the best one for both train and validation sets, with NMI at 0.64 and 0.75, ARI at 0.63 and 0.74, and Acc at 0.79 and 0.83, respectively. This shows that if the latent dimension is large enough, increasing the number of epochs makes the model learn better up to a certain threshold, after which it may stop improving or even worsen. In both Tables 4.6 and 4.7, as the number of epochs and latent dimensions increased, the performance of the clustering model in terms of ARI and NMI initially improved but declined when the latent dimensions reached 128. However, this observation does not hold for accuracy, as Table 4.6 exhibited the same trend as discussed previously, whereas, in Table 4.7, accuracy increased alongside the latent dimensions.

Considering Experiments 4 and 5, we observe that they each have 400 epochs but different latent dimensions (32 and 128, respectively). The performance of Experiment 4 is somewhat similar, if not better, than that of Experiment 5 in all the metrics except for the K value, where Experiment 4 performed better in Table 4.6, resulting in number of clusters close to 4 (Final K = 4) compared to the original number (Original K = 5). While in Table 4.7, Experiment 4 had a better performance than Experiment 5, the only exception being in terms of accuracy for both training and validation sets. This suggests that the configuration was not as effective in clustering and classification tasks. This shows that the latent dimension and no of epochs act as a tunable hypermeter specific to the dataset enabling the model to capture complicated data patterns, resulting in enhanced clustering performance.

In our case, with a latent dimension of 64 and 400 epochs, Experiment 2 had the best clustering performance overall. This combination of latent dimension and epochs gives the ideal balance between the model's capacity to learn complicated patterns, training duration, and convergence.

Upon analyzing the training and validation results for the five experiments, with three replicates each, we did not observe indications of overfitting or underfitting in most cases. In most experiments, the training and validation metrics are close, indicating a good balance and suggesting that our results do not suffer from overfitting or underfitting. The only exception is in the case of Experiment 1 where the validation accuracy value was lower than the train (Table 4.7). Experiment 1 demonstrates lower performance metrics, especially when compared to other experiments. This could be attributed to underfitting, possibly caused by an inadequate number of training epochs. Experiments 2, 3, and 4 exhibits relatively stable performance across training and validation sets, with only minor differences in the metrics. This suggests that these models are better at generalizing to unseen data and have achieved a better balance between overfitting and underfitting. Lastly, Experiment 5 has a lower NMI and ARI on both the training and validation sets compared to other experiments, suggesting that the model may be underfitting the data. However, the overall performance is still good, which indicates that overfitting is not a major problem in these models.

In conclusion, a suitable latent dimension and a number of epochs are required for effective clustering performance. The right latent size typically enables the model to capture and learn complex patterns in the data, while the right number of epochs allows the model to learn effectively without overfitting or underfitting. Finding the best combination of these two factors is key to achieving the best clustering performance. Altogether, the improved results indicate that the embeddings generated using ViT-MAE are of higher quality and more suitable for clustering fish images from the Fish4Knowledge dataset, independent of the image enhancement methods used compared to the earlier MoCo embeddings. DeepDPM gave variable results with each run, so it may be necessary to run the clustering algorithm multiple times to obtain valuable clusters under each setting. In our experiments we ran the DeepDPM model for three times for every setting and reported the average result along with its standard deviation.

4.8 Visualizing the clusters

Figure 4.9 presents a visualization tool that allows for the examination of clustered data about various fish species. Upon analysis of the clustered outputs, our model did not find the correct clusters; however, it was capable enough to identify the major fish species in the dataset to a certain extent. The three major clusters correspond closely to the three most common fish in this dataset. The model primarily struggled to cluster fish species with lower frequencies accurately. The two least common fish were grouped together with Chromis chrysura in the fourth cluster.

A more comprehensive understanding of these findings can be gleaned from the stacked bar graph in Figure 4.14, which displays the count of each fish species within the respective clusters. This visualization is derived from the most favorable experiment we conducted, which utilized ViT-MAE embeddings of the ESRGAN-enhanced dataset with a latent dimension of 64 and a training duration of 400 epochs.

Cluster 3 is the largest having 859 images, with Chromis chrysura constituting 82% of them. The remaining proportions are comprised of Myripristis kuntee (8%), Hemigymnus fasciatus (5%), Amphiprion clarkii (3%), and Chaetodon lunulatus (1%). Cluster 2 contains the least number of images, totaling 22. It is predominantly composed of Chaetodon lunulatus (14 images), followed by Amphiprion clarkii (5 images), Hemigymnus fasciatus (2 images), and a minor presence of Chromis chrysura (1 image). Cluster 0 is nearly exclusively Chaetodon lunulatus (99.79%), with an insignificant presence of Chromis chrysura. Cluster 1 primarily consists of Amphiprion clarkii (98.72%), accompanied by a small percentage of Chaetodon lunulatus and an even smaller percentage of Hemigymnus fasciatus. Cluster 4 mainly comprises Amphiprion clarkii (98.02%), with minor proportions of Chaetodon lunulatus and Hemigymnus fasciatus.

Upon examining the final clustering results, it is important to recognize that this model should be applied cautiously, as it primarily distinguishes species with high counts while failing to recognize the rare-group species, which were combined into single clusters. To address this issue, we hypothesize that the clustering model or other approaches could be further applied to the combined clusters to subdivide them and identify the rare species. Also, it is worth noting that the clustering results are influenced by the choice of embeddings, latent dimensions, and number of epochs utilized in the training

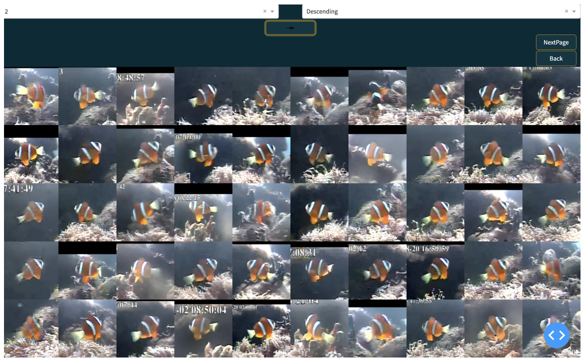


Figure 4.9: Visualization dashboard to view the clusters - Cluster 1

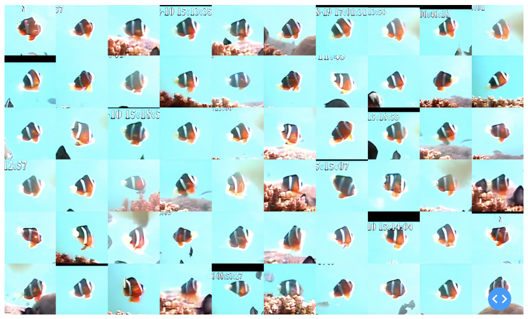


Figure 4.10: Cluster 4

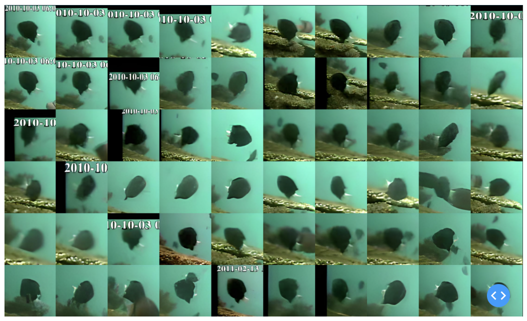


Figure 4.11: Cluster 3

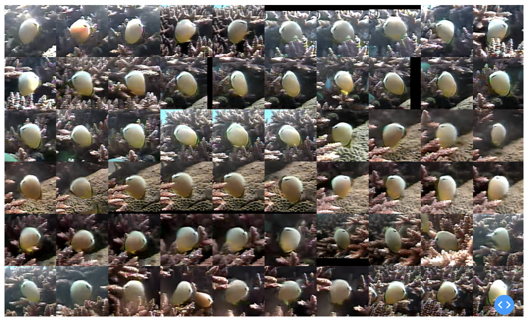


Figure 4.12: Cluster 0

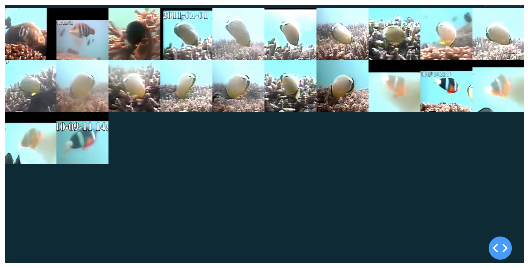


Figure 4.13: Cluster 2

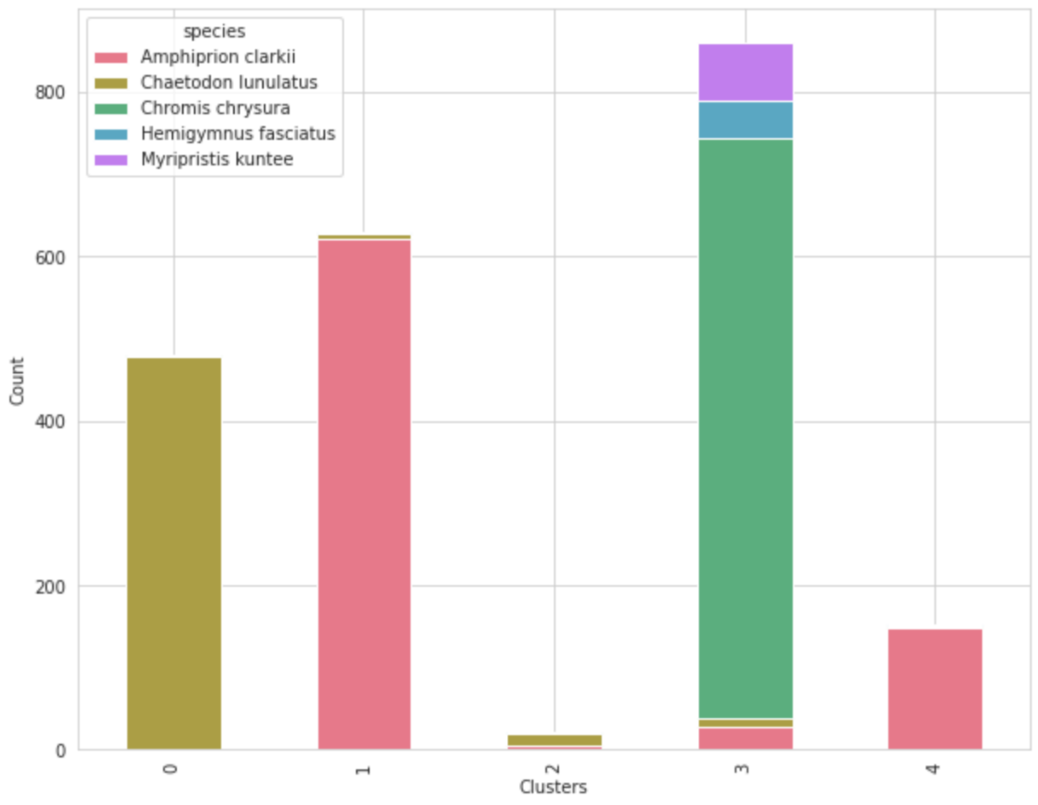


Figure 4.14: Fish species count in each cluster

Chapter 5

Conclusion and Future work

5.1 Conclusion

In this analysis, non-parametric and unsupervised clustering algorithm models were tested on the Fish4Knowledge dataset to see how well they clustered fish images using different embeddings with and without image enhancement. The goal was to investigate the potential for unsupervised clustering methods to separate fish by species and reduce the time needed to label datasets for supervised models. After reviewing our final outcome on the Fish4Knowledge dataset, it is worth noting that achieving a close approximation, rather than an exact number of clusters, may be considered acceptable if the resulting clusters demonstrate strong consistency with the actual species classification.

The original Fish4Knowledge dataset had low-resolution images from software that found and tracked fish, which made clustering less accurate. As expected, after enhancing the images using the Real-ESRGAN deep learning approach, the clustering results improved slightly, resulting in higher accuracy rates for all three evaluation metrics (NMI, ARI, and Acc). Our results showed that the initial pre-trained MoCo model had poor performance. After applying the image enhancement, the performance was slightly improved, and the final number of clusters became a bit more consistent. From our findings, the MoCo model trained from scratch on the fish4k dataset outperformed the ImageNet pre-trained model in terms of NMI, ARI, and accuracy. The image improvement on top of it also led to better performance; in the end, the number of clusters averaged between 3 and 4. In the end, the results of the study suggest that the embeddings made with ViT-MAE are better suited to grouping fish images from the Fish4Knowledge dataset. Overall, the combination of ViT-MAE embeddings with the Real-ESRGAN image enhancement improved the mean values of NMI, ARI and accuracy values from 58, 55, 77 to 69, 67, 80. Our results suggest it is possible to achieve good clustering results by enhancing the picture's quality and employing embeddings generated by a suitable self-supervised model. Our study presented the results by calculating the mean of three separate runs for each experimental setting and subsequently comparing these values using the standard deviation.

It is important to note that we did not conduct an in-depth statistical analysis to assess the validity of our findings, which may necessitate further investigation to confirm the robustness of the results.

In summary, we found that unsupervised clustering models (DeepDPM) [79] could be a promising alternative to supervised deep learning for classifying fish images. We also discovered that image enhancement techniques helped improve clustering performance. Additionally, we observed that embeddings from ViT-MAE outperformed those obtained from MoCo on Fish4Knowledge dataset. Later, we wish to apply our model on the remaining species of fish4k dataset as well.

Our study demonstrates that unsupervised clustering models hold promise for accelerating the process of fish species labeling by utilizing the unsupervised clustering algorithm DeepDPM and enhancing its performance through image improvement techniques, suitable embeddings, and ideal latent dimensions. These models could shorten the time needed for human labeling by grouping the bulk of fish species into discrete groups, making labeling easy. However, more research and validation are required to prove that these models successfully reduce the burden associated with manual labeling.

5.2 Limitations and Future work

Our goal with this work was to investigate whether unsupervised clustering methods could be applied to group fish images into clusters of species without the need for manual labeling of species. As such, there are several limitations to our results and many avenues that require further exploration.

The first problem is the lack of enough training data. Deep learning models trained with a large amount of data can perform more effectively and prevent overfitting. [47]. One alternative to this is to use self-supervised pretraining. We generally observe that the performance of deep learning models on supervised tasks is improved by unsupervised pretraining, which leads to improved initialization, regularization, and feature extraction[33]. DeepDPM paper for instance, uses self-supervised pretraining using models such as MoCo and then performs clustering.

In our particular use case, we observed that the embeddings obtained from pretraining using the ViT-MAE model were more effective than those obtained from the MoCo model. Several self-supervised learning approaches have been proposed recently, including I-JEPA [41], which is a non-generative approach for self-supervised learning from images that predicts the representations of different parts of an image from a single context part. It is based on JEPA, a general framework for self-supervised learning that captures dependencies between two inputs [63] [61]. In addition to I-JEPA, other approaches such as BeiTv2, SimMIM, DINO, and MoCo v3 have also been proposed. BeiTv2 involves masked image modeling with vector-quantized representations [75], while SimMIM utilizes a ViT encoder with both masked and non-masked patches as inputs [96]. DINO, which stands for self-distillation with no labels, is a self-supervised learning approach that predicts the output of a teacher network using a standard cross-entropy loss [14]. Meanwhile, MoCo v3 has a structure similar to SimCLR but differs in its momentum encoder [18].

Another problem that arises is due to the absence of high-resolution images in the Fish4Knowledge dataset because fish are usually a small part of the total camera image. It is well known that the quality of images has a crucial influence on the performance of deep learning models. It can affect the models' ability to generalize and maintain robustness. The presence of low-quality images in the training dataset may introduce biases or outliers, which can hinder the model's performance on new, unseen data. This phenomenon was observed in the results of the clustering analysis conducted on the Fish4Knowledge dataset, where images with poor quality contained less informative data and were more challenging to classify using unsupervised learning. Factors such as noise or blur in images can obscure important object features, making it difficult for deep-learning models to detect or classify them [89]. It is important to note that low-quality images may not accurately represent the true distribution of real-world data, causing the models to perform poorly on unseen data. Examples of factors that can cause image degradation include compression artifacts or low resolution, which can alter the shapes or colors of objects and make them appear different from what the model learned during training [28] [77]. Although ESRGAN was utilized to enhance the quality of fish images in the Fish4Knowledge dataset, there are numerous other image enhancement techniques that can also be employed. One such technique is Swin2SR, an improved version of SwinIR, which incorporates Swin Transformer v2 layers to address issues such as training instability, resolution gaps between pretraining and fine-tuning, and data requirements [67]. Swin2SR has demonstrated state-of-the-art performance in classical, lightweight, and real-world image super-resolution tasks [67]. SwinIR, on the other hand, is an image restoration tool based on the Swin Transformer architecture, which comprises shallow and deep feature extraction modules and a high-quality image reconstruction module [66].

In some of our experiments we successfully obtained five clusters, which matched the original number of clusters in the dataset. However, these clusters grouped infrequent species together and were not fully coherent enough to effectively classify the fish according to their respective species. It is essential to note that these results, although promising, have been obtained from a limited dataset. Also, the performance of DeepDPM exhibited a slight variability across different runs; this variability, although it is less, represents a limitation of the DeepDPM method, as it may require knowing the suitable configuration so that the data can be clustered easily. After looking at the clusters, it is essential to consider that there may be instances where it needs to be clarified whether the clustering was performed based on specific attributes such as color, shape, or size. So, to validate the effectiveness of the proposed clustering model in real-world applications, further testing on more extensive and diverse datasets is necessary. Also, it is necessary to check how well the model works on new, unlabeled datasets to see if the clusters it makes help cut down on labeling time and classify fish by species.

In conclusion, the clustering results from this work are promising and suggest that unsupervised learning models can be used to classify fish by species, which will cut down on the time needed to label them. Nevertheless, it is crucial to continue to refine the models, and experiment with different enhancement techniques and embeddings, and evaluate their performance on a broader range of datasets to fully understand their potential impact on the fish species labeling process.

Unsupervised learning techniques may potentially aid in accelerating the process of labeling fish species and, to some extent, assist in identifying rare or previously unobserved species in new datasets. Since these methods do not use already-made labels, they can find unique patterns in outlier species and group them reasonably well. This capability facilitates the efforts of researchers and professionals in identifying and tracking uncommon species that might otherwise be overlooked or misidentified through traditional supervised learning techniques. DeepDPM clustering has the potential to assist in the identification of uncommon fish; however, the current results suggest that further refinement is required to distinguish rarer species when dealing with highly unbalanced data properly. By solving this issue, unsupervised learning approaches can become even more important discovery and monitoring tools for elusive or uncommon fish species.

Bibliography

[1] Unsupervised deep embedding for clustering analysis, 2019.

[2] Efficient and precise single-cell reference atlas mapping with symphony. 2020.

[3] 2022. [online]. available: https://nlai.blue/the-importance-of-monitoring-marine-ecosystems-for-sustainable-ocean-resource-use/. [accessed: 08- jul- 2022]., 2022.

[4] Fish recog ”fish recognition ground-truth data”, homepages.inf.ed.ac.uk, 2022. [online]. available: https://homepages.inf.ed.ac.uk/rbf/fish4knowledge/groundtruth/recog/. [accessed: 09- jul- 2022]., 2022.

[5] Pixel intensity - an overview. `https://www.sciencedirect.com/topics/computer-science/pixel-intensity`, 2022.

[6] What does intensity mean in image processing? `https://sage-answer.com/what-does-intensity-mean-in-image-processing/`, 2022.

[7] Mutasem K Alsmadi and Ibrahim Almarashdeh. A survey on fish classification techniques. *Journal of King Saud University-Computer and Information Sciences*, 2020.

[8] Charles E Antoniak. Mixtures of dirichlet processes with applications to bayesian nonparametric problems. *The Annals of Statistics*, pages 1152–1174, 1974.

[9] Kameswari Devi Ayyagari, Christopher Whidden, Corey Morris, and Joshua Barnes. Towards low cost automated monitoring of life below water to de-risk ocean-based carbon dioxide removal and clean power. In *NeurIPS 2022 Workshop: Tackling Climate Change with Machine Learning*, 2022.

[10] Atılım Güneş Baydin, Barak A Pearlmutter, Alexey Andreyevich Radul, and Jeffrey Mark Siskind. Automatic differentiation in machine learning: a survey. *The Journal of Machine Learning Research*, 18(1):5595–5637, 2018.

[11] Lawrence R Beerkircher, Freddy Arocha, Alex Barse, Eric Prince, Victor Restrepo, Joseph Serafy, and Mahmood Shivji. Effects of species misidentification on population assessment of overfished white marlin tetrapturus albidus and roundscale spearfish t. georgii. *Endangered Species Research*, 9:81–90, 2009.

[12] Jacob Bergman, Kevin Bierlich, Robert Schick, and David Johnston. Improved accuracy for automated counting of a fish in video using deep learning. *Frontiers in Marine Science*, 8:658135, 2021.

[13] BGU-CS-VIL. Deepdpm. `https://github.com/BGU-CS-VIL/DeepDPM`, 2022.

[14] Mathilde Caron, Hugo Touvron, Ishan Misra, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. *arXiv preprint arXiv:2104.14294*, 2021.

[15] Jason Chang and John W Fisher III. Parallel sampling of dp mixture models using sub-cluster splits. *Advances in Neural Information Processing Systems*, 26, 2013.

[16] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. *arXiv preprint arXiv:2002.05709*, 2020.

[17] Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. Improved baselines with momentum contrastive learning. *arXiv preprint arXiv:2003.04297*, 2020.

[18] Xinlei Chen, Saining Xie, and Kaiming He. An empirical study of training self-supervised vision transformers. *arXiv preprint arXiv:2104.02057*, 2021.

[19] Yu-Cheng Chen, Chih-Chung Chen, and Chih-Hung Lin. An automated fish species classification system using improved alexnet and transfer learning. In *2017 IEEE International Conference on Consumer Electronics-Taiwan (ICCE-TW)*, pages 1–2. IEEE, 2017.

[20] Yunpeng Chen, Xiaohang Zhan, Xinyu Gong, Siyuan Qiao, and Changshui Zhang. Unsupervised learning of visual representations by solving jigsaw puzzles. *CoRR*, abs/2106.05232, 2021.

[21] Adam Coates, Andrew Ng, and Honglak Lee. An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 215–223. JMLR Workshop and Conference Proceedings, 2011.

[22] Hugo Costa, Giles M Foody, S'ılvia Jim'enez, and Lu'ıs Silva. Impacts of species misidentification on species distribution modeling with presence-only data. *International Journal of Geo-Information*, 4(4):2496–2518, 2015.

[23] Lopes-Silva Tarcísio de Melo; Silva-Júnior Antônio Carlos; Santos-Silva Edson Nascimento; Pompeu Paulo dos Santos. Evaluation of three methods for manually counting fish in dam turbines: numerical counting versus intersection counting versus qualitative counting with a correction factor applied to each method hydrobiologia. *Hydrobiologia*, 848(18):4157–4168, 2021.

[24] MP Dekar, LR Brown, JA Hobbs, and et al. Fish misidentification and potential implications to monitoring within the san francisco estuary, california. *Fisheries Research*, 9(2):467–474, 2018.

[25] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database, 2009.

[26] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009.

[27] Li Deng. The mnist database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine*, 29(6):141–142, 2012.

[28] Samuel F Dodge and Lina J Karam. Understanding how image quality affects deep neural networks. *arXiv preprint arXiv:1604.04004*, 2016.

[29] Alves-Pereira Alessandra; Silva-Júnior Antônio Carlos; Santos-Silva Edson Nascimento; Pompeu Paulo dos Santos. Automated detection, classification and counting of fish in fish passes using deep learning models and computer vision techniques. *Frontiers in Marine Science*, 8:823173, 2021.

[30] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[31] Salim Dridi. Supervised learning - a systematic literature review. *Cornell University*, 2021.

[32] Salim Dridi. Unsupervised learning - a systematic literature review. *ResearchGate*, 2022.

[33] D. Erhan, Y. Bengio, A. Courville, P. Manzagol, P. Vincent, and S. Bengio. Why does unsupervised pre-training help deep learning? *Journal of Machine Learning Research*, 11, 2010.

[34] Hugging Face. Transformers. `https://github.com/huggingface/transformers`, 2022.

[35] Thomas S Ferguson. A bayesian analysis of some nonparametric problems. *The Annals of Statistics*, pages 209–230, 1973.

[36] Mohamad Mostafa M Fouad, Hossam M Zawbaa, Tarek Gaber, Vaclav Snasel, and Aboul Ella Hassanien. A fish detection approach based on bat algorithm. In *International Conference on Advanced Intelligent Systems and Informatics*, pages 273–283. Springer, 2016.

[37] Wouter Van Gansbeke, Simon Vandenhende, Stamatios Georgoulis, Marc Proesmans, and Luc Van Gool. Scan: Learning to classify images without labels, 2020.

[38] Jonatan Garc'ıa-Guti'errez, Ana Garc'ıa-Serrano, Bel'en Ru'ız-Mezcua, and Fernando Mart'ınez-Santiago. Evaluation metrics for unsupervised learning algorithms. *arXiv preprint arXiv:1905.05667*, 2019.

[39] Eva Garcia-Vazquez, Gonzalo Machado-Schiaffino, Daniel Campo, and Francis Juanes. Species misidentification in mixed hake fisheries may lead to overexploitation and population bottlenecks. *Fisheries Research*, 114:52–55, 2012.

[40] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *arXiv preprint arXiv:1406.2661*, 2014.

[41] Priya Goyal, Mathilde Caron, Benjamin Lefaudeux, and Yann LeCun. Self-supervised learning from images with a joint embedding predictive architecture, 2022.

[42] Hayit Greenspan and Tanveer Syeda-Mahmood. Parametric and non-parametric clustering for segmentation. In *Biomedical Image Processing*, pages 227–250. Springer, 2010.

[43] J S Griffith. Estimation of the age-frequency distribution of stream-dwelling trout by underwater observation. *The Progressive Fish-Culturist*, 43(1):51–53, 1981.

[44] Douglas R Hatch, Mark Schwartzberg, and Peter R Mundy. Estimation of pacific salmon escapement with a time-lapse video recording technique. *North American Journal of Fisheries Management*, 14(3):626–635, 1994.

[45] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9729–9738, 2020.

[46] Tong He, Bo Chen, Lei Wang, Zhiqiang Zhang, and Liang Zhang. Masked autoencoders are scalable vision learners. *arXiv preprint arXiv:2111.06377*, 2021.

[47] Ren S. Sun J. He K., Zhang X. Deep residual learning for image recognition. *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[48] Geoffrey E Hinton and Ruslan R Salakhutdinov. Reducing the dimensionality of data with neural networks. *science*, 313(5786):504–507, 2006.

[49] Geoffrey E Hinton and Richard S Zemel. Autoencoders, minimum description length and helmholtz free energy. In *Advances in Neural Information Processing Systems*, 1993.

[50] J. J. Hull. A database for handwritten text recognition research. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(5):550–554, 1994.

[51] Abdelhameed Ibrahim, Ali Ahmed, Sherif Hussein, and Aboul Ella Hassanien. Fish image segmentation using salp swarm algorithm. In Aboul Ella Hassanien, Mohamed F. Tolba, Mohamed Elhoseny, and Mohamed Mostafa, editors, *The International Conference on Advanced Machine Learning Technologies and Applications (AMLTA2018)*, pages 42–51, Cham, 2018. Springer International Publishing.

[52] S. S. Kadam and S. S. Kulkarni. Unsupervised learning - a systematic literature review. *International Journal of Computer Applications*, 182(30):1–6, 2021.

[53] Vishnu Kandimalla, Matt Richard, Frank Smith, Jean Quirion, Luis Torgo, and Chris Whidden. Automated detection, classification and counting of fish in fish passages with deep learning. *Frontiers in Marine Science*, page 2049, 2022.

[54] Harpreet Kaur. How to evaluate unsupervised learning models. *Towards Data Science Blog Post (https://towardsdatascience.com/how-to-evaluate-unsupervised-learning-models-3aa85bd98aa2)*, 2020.

[55] Asifullah Khan, Aisha Khanum, and Abdul Rauf Baig. Automatic fish species classification using deep convolutional neural networks: a comprehensive study. *Wireless Personal Communications*, 114(1):579–595, 2020.

[56] Diederik P Kingma and Max Welling. An introduction to variational autoencoders. *arXiv preprint arXiv:1906.02691*, 2019.

[57] Joseph E Kirsch, Robert Feeney, Anne Goodbla, Catherine Hart, Zachary J Jackson, Andrea Schreier, and Ryan Smith. The first record of large-scale loach occurrence in the united states. *Journal of Fish and Wildlife Management*, 9:246–254, 2018.

[58] Kristian Muri Knausgård, Arne Wiklund, Tonje Knutsen Sørdalen, Kim Tallaksen Halvorsen, Alf Ring Kleiven, Lei Jiao, and Morten Goodwin. Temperate fish detection and classification: a deep learning based approach. *Applied Intelligence*, 52(6):6988–7001, 2022.

[59] Huei-Meei Ko, Wei-Jen Wang, Tai-Sheng Chiu, Meng-Hsien Lee, Ming-Yih Leu, Kuan-Ting Chang, Wei-Jen Chen, and Kwang-Tsao Shao. Evaluating the accuracy of morphological identification of larval fishes by applying dna barcoding. *PLoS ONE*, 8(1):e53451, 2013.

[60] Sotiris B Kotsiantis. Supervised machine learning: A review of classification techniques. *Informatica (Slovenia)*, 31(3):249–268, 2007.

[61] Yann LeCun. A vision to make ai systems learn and reason like humans, 2019.

[62] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[63] Yann LeCun, Ben Dickson, Ben Dickson, Ben Dickson, Ben Dickson, Ben Dickson, Ben Dickson, Ben Dickson, Ben Dickson, Ben Dickson, and Ben Dickson. Meta's yann lecun on his vision for human-level ai: A conversation with techtalks founder ben dickson (part ii). *TechTalks*, 3(1):1–18, March 2022.

[64] Christian Ledig, Lucas Theis, Ferenc Huszar, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew P Aitken, Alykhan Tejani, Johannes Totz, and Zehan Wang. Photo-realistic single image super-resolution using a generative adversarial network. *arXiv preprint arXiv:1609.04802*, 2016.

[65] Yuxin Li, Jiaqi Liang, Zhiyong Zhang, and Xuefeng Wang. Exploring and comparing unsupervised clustering algorithms. *Journal of Physics: Conference Series*, 1668(3):032012, 2020.

[66] Jingyun Liang, Jiezhang Cao, Huajun Sun, Xiaodong Xu, Zhengjun Zhang, and Liqing Zhang. Swinir: Image restoration using swin transformer. *arXiv preprint arXiv:2108.10257*, 2021.

[67] Jingyun Liang, Jiezhang Cao, Huajun Sun, Xiaodong Xu, Zhengjun Zhang, and Liqing Zhang. Swin2sr: Swinv2 transformer for compressed image super-resolution. *arXiv preprint arXiv:2209.11345*, 2022.

[68] Lightly. Lightly. https://docs.lightly.ai/self-supervised-learning/examples/moco.html, 2023.

[69] Peter B Marko, Shing C Lee, Aaron M Rice, John M Gramling, Thomas M Fitzhenry, Justin S McAlister, Glenn R Harper, and Amy L Moran. Mislabelling of a depleted reef fish. *Nature*, 430(6996):309–310, 2004.

[70] Brett T McClintock, Larissa L Bailey, Kenneth H Pollock, and Theodore R Simons. Unmodeled observation error induces bias when inferring patterns and dynamics of species occurrence via aural detections. *Ecology*, 91(8):2446–2454, 2010.

[71] Leland McInnes and John Healy. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018.

[72] Jessica L Metcalf, Victoria L Pritchard, Sarah M Silvestri, Jill B Jenkins, Jeffrey S Wood, David E Cowley, R Paul Evans, Dennis K Shiozawa, and Andrew P Martin. Across the great divide: genetic forensics reveals misidentification of endangered cutthroat trout populations. *Molecular Ecology*, 16(21):4445–4454, 2007.

[73] Jini Mol and Albin Jose. Fish species classification using optimized deep learning model. *International Journal of Advanced Computer Science and Applications*, 13(9):798–804, 2021.

[74] Y Nagashima and T Ishimatsu. A morphological approach to fish discrimination. In *IAPR Workshop on Machine Vision Applications*, pages 306–309, 1998.

[75] Zhiliang Peng, Li Dong, Hangbo Bao, Qixiang Ye, and Furu Wei. Beit v2: Masked image modeling with vector-quantized visual tokenizers. *arXiv preprint arXiv:2208.06366*, 2022.

[76] James T Peterson and Craig Paukert. Data conversion. In *Standard Sampling Methods for North American Freshwater Fishes*, pages 195–216. American Fisheries Society, 2009.

[77] Hong Mostofi Yasamin Sen Pradeep Prashnani, Ekta Cai. Deeper image quality transfer: Training low-memory neural networks for blind image quality assessment. *arXiv preprint arXiv:1808.05577*, 2018.

[78] Freitas M.H.G Rodrigues, M.T.A. and F.L.C Pádua. Evaluating cluster detection algorithms and feature extraction techniques in automatic classification of fish species. *Pattern Analysis and Applications*, 2014.

[79] Meitar Ronen, Shahaf E Finder, and Oren Freifeld. Deepdpm: Deep clustering with an unknown number of clusters. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9861–9870, 2022.

[80] Shoffan Saifullah. K-means and morphological approach on image segmentation for fish detection. In *2022 19th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON)*, pages 1–4, 2022.

[81] Alzayat Saleh, Marcus Sheaves, and Mostafa Rahimi Azghadi. Computer vision and deep learning for fish classification in underwater habitats: A survey. *arXiv preprint arXiv:2203.06951*, 2022.

[82] Sarika. Underwater fish species recognition using deep learning techniques. In *2019 6th International Conference on Signal Processing and Integrated Networks (SPIN)*, pages 7–8. IEEE, 2019.

[83] Daniel J Schill and James A Lamansky, Jr. The ability of southwest idaho anglers to identify five species of trout. *Idaho Fish and Game Report No. 00-12*, 1999. see Supplemental Material, Reference S3.

[84] Fisheries Inventory Section. Fish collection methods and standards: Version4 [pdf file], 1998.

[85] Faisal Shafait, Ajmal Mian, Mark Shortis, Bernard Ghanem, Phil F Culverhouse, Duane Edgington, Danelle Cline, Mehdi Ravanbakhsh, James Seager, and Euan S Harvey. Fish identification from videos captured in uncontrolled underwater environments. *ICES Journal of Marine Science*, 73(10):2737–2746, 2016.

[86] Concetto Spampinato, Daniela Giordano, Rosario Di Salvo, Yun-Heh Jessica Chen-Burger, Robert B Fisher, and Gayathri Nadarajan. Automatic fish classification for underwater species behavior understanding. In *Proceedings of the first ACM international workshop on Analysis and retrieval of tracked events and motion in imagery streams*, pages 45–50. ACM, 2010.

[87] Frank Storbeck and Berent Daan. Fish species recognition using computer vision and a neural network. *Fisheries Research*, 51(1):11–15, 2001.

[88] Y. Sun and X. Luo. Algorithm of adaptive fast clustering for fish swarm color image segmentation. In *2019 IEEE 4th International Conference on Cloud Computing and Big Data Analysis (ICCCBDA)*, 2019.

[89] Hongyi Tang, Neel Joshi, and Ashish Kapoor. Deep learning network for blind image quality assessment. In *2014 IEEE International Conference on Image Processing (ICIP)*, pages 377–381. IEEE, 2014.

[90] Wouter Van Gansbeke, Simon Vandenhende, Stamatios Georgoulis, Marc Proesmans, and Luc Van Gool. Scan: Learning to classify images without labels. In *European conference on computer vision*, pages 268–285. Springer, 2020.

[91] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

[92] Xintao Wang. Real-esrgan. `https://github.com/xinntao/Real-ESRGAN`, 2022.

[93] Xintao Wang, Liangbin Xie, Kaiyang Yu, Chao Dong, and Chen Change Loy. Real-esrgan: Training real-world blind super-resolution with pure synthetic data. *arXiv preprint arXiv:2107.10833*, 2021.

[94] Xintao Wang, Ke Yu, Shixiang Wu, Jinjin Gu, Yihao Liu, Chao Dong, Yu Qiao, and Chen Change Loy. Esrgan: Enhanced super-resolution generative adversarial networks. *arXiv preprint arXiv:1809.00219*, 2018.

[95] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms, 2017. cite arxiv:1708.07747Comment: Dataset is freely available at https://github.com/zalandoresearch/fashion-mnist Benchmark is available at http://fashion-mnist.s3-website.eu-central-1.amazonaws.com/.

[96] Zhenda Xie, Zheng Zhang, Yue Cao, Yutong Lin, Jianmin Bao, Zhuliang Yao, Qi Dai, and Han Hu. Simmim: A simple framework for masked image modeling. *arXiv preprint arXiv:2111.09886*, 2021.

[97] Bo Yang, Xiao Fu, Nicholas D Sidiropoulos, and Mingyi Hong. Towards k-means-friendly spaces: Simultaneous deep learning and clustering. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 3861–3870. JMLR. org, 2017.

[98] Hong Yao, Qingling Duan, Daoliang Li, and Jianping Wang. An improved k-means clustering algorithm for fish image segmentation. *Mathematical and Computer Modelling*, 58(3):790–798, 2013. Computer and Computing Technologies in Agriculture 2011 and Computer and Computing Technologies in Agriculture 2012.

[99] Yifan Zhang, Xiang Li, Yijun Wang, and Xiang Zhang. Deep clustering: A comprehensive survey. *arXiv preprint arXiv:2210.04142*, 2022.

[100] Tingting Zhao, Zifeng Wang, Aria Masoomi, and Jennifer G Dy. Streaming adaptive nonparametric variational autoencoder. *arXiv preprint arXiv:1906.03288*, 2019.