Image Retrieval Systems
Based on Embeddings for Trustworthy AI

A thesis

submitted in fulfilment

of the requirements for the degree

of

Master of Science (Research) in Artificial Intelligence

at

The University of Waikato

by

Zane N Neave

THE UNIVERSITY OF
WAIKATO
Te Whare Wānanga o Waikato

2025

Acknowledgments

I would like to express my sincere gratitude to my advisor, Professor Bernhard Pfahringer, for their invaluable guidance, encouragement, and support throughout this research. Their insights and expertise have been instrumental in shaping this work.

Furthermore, I acknowledge The University of Waikato for providing the necessary resources for conducting this research.

I would like to express my sincere gratitude to Marine AI for their support and insights, which have enriched this research. I also appreciate the contributions from Syos Aerospace.

Lastly, I would like to thank the broader academic community whose research and contributions have inspired and guided this study.

Abstract

Embedding-based image retrieval systems have become essential for applications that require high accuracy, computational efficiency, and adaptability, particularly in fields like healthcare and agriculture. These systems utilize deep learning models to generate high-dimensional feature embeddings, enabling scalable and precise image retrieval. This study evaluates two distinct retrieval pipelines: the Domain-Specific Retrieval Pipeline (DSRP) and the Pre-trained Feature Extraction Pipeline (PMRP). The comparative analysis, conducted across multiple datasets including medical imaging, agricultural analysis, and general object retrieval, highlights the advantages and trade-offs between these approaches.

The results demonstrate that PMRP achieves consistently higher retrieval accuracy across all datasets, with near-perfect performance in tasks like leaf disease detection and general object retrieval. In contrast, DSRP, while benefiting from domain-specific adaptation, does not surpass PMRP in overall performance and exhibits variability depending on dataset characteristics. Despite its domain-tuned architecture, DSRP showed increased computational complexity without a proportional increase in accuracy. Additionally, PMRP maintains more stable inference times, making it a preferable option for real-time applications.

Beyond retrieval accuracy, this study explores challenges related to dataset diversity, bias mitigation, explainability, and adversarial robustness. The findings underscore the importance of balancing retrieval precision with computational feasibility, particularly when deploying systems in high-stakes environments. Future research should focus on enhancing dataset representation, developing fairness-aware learning mechanisms, improving model transparency through explainable AI techniques, and strengthening adversarial defenses to ensure the security and reliability of embedding-based retrieval systems. By addressing these challenges, such systems can evolve into more efficient, interpretable, and ethically responsible solutions for real-world applications across various domains.

Contents

Chapter 1

Introduction

Machine learning (ML) has emerged as one of the most transformative technologies of the 21st century, revolutionizing numerous domains by enabling high-speed, high-accuracy processing of vast amounts of information [1]. This capability has had a significant impact on areas such as medical diagnostics, automated decision-making, and image processing, offering new opportunities for efficiency, precision, and cost reduction [2]. The advent of deep learning architectures, particularly Convolutional Neural Networks (CNNs), has further enhanced these capabilities, making it possible to extract meaningful patterns and insights from complex datasets [3].

However, as artificial intelligence (AI) systems are increasingly integrated into critical decision-making processes, concerns regarding their trustworthiness, interpretability, and ethical integrity have gained prominence [4]. Many AI models function as "black boxes," providing highly accurate predictions but lacking transparency in their decision-making processes [5]. This opacity raises fundamental questions about reliability, fairness, and accountability, particularly in domains where AI-driven decisions can have life-altering consequences, such as in healthcare, autonomous vehicles, and financial systems [6]. Furthermore, biases inherent in training datasets can lead to discriminatory outcomes, making it imperative to develop AI systems that are robust against bias and adversarial manipulation [7].

To address these challenges, a new paradigm in AI-driven image processing has emerged: embedding-based image retrieval systems. These systems transform images into high-dimensional vector representations—known as embeddings, that encode essential features, allowing for rapid and accurate image retrieval based on similarity metrics [8]. Unlike traditional pixel-based approaches, embedding-based retrieval offers improved generalization by accommodating variations in lighting, orientation, and background noise [9]. Such systems have significant implications for real-world applications, particularly in medical imaging, agriculture, and autonomous navigation, where interpretability and reliability are of utmost importance.

This dissertation introduces an advanced embedding-based image retrieval framework that enhances transparency and accountability in AI-assisted image processing. The proposed system leverages state-of-the-art deep learning methodologies, including CNN-based feature extraction and clustering algorithms, to ensure high accuracy while addressing concerns related to interpretability

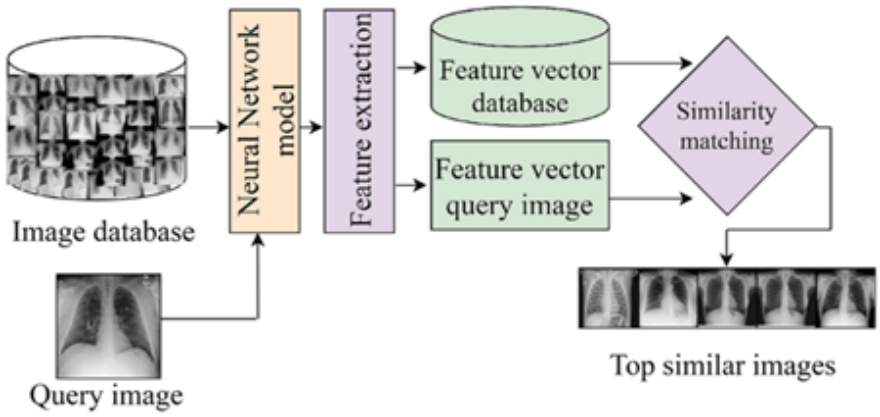

Figure 1.1: Image Retrieval Flowchart based on Embedding Extraction

and fairness [10]. The framework has been extensively evaluated across multiple domains to ensure its scalability, resilience, and real-world applicability.

One of the most critical applications of this system lies in healthcare, where AI-driven image retrieval can significantly enhance disease diagnosis and treatment planning [3]. For instance, deep learning models can assist in detecting early stage tumours, identifying retinal abnormalities, and diagnosing cardiovascular diseases with a high degree of accuracy [6]. By incorporating explainable AI (XAI) techniques, clinicians can interpret the reasoning behind the model's predictions, fostering trust and reliability [7].

Beyond healthcare, embedding-based image retrieval also plays a crucial role in agriculture and environmental monitoring. AI-powered systems can differentiate between crops and weeds, optimize pesticide application, and improve overall agricultural productivity [7]. Additionally, autonomous navigation systems rely on image retrieval to identify obstacles, understand surroundings, and make real-time decisions, enhancing safety and efficiency [9].

This research not only advances technical capabilities in AI-driven image retrieval but also contributes to the broader discourse on responsible AI development. The integration of interpretability techniques ensures that AI systems remain transparent, fair, and aligned with human values, promoting ethical adoption across industries [10]. By addressing key challenges such as bias mitigation, adversarial robustness, and human-AI interaction, this dissertation provides a blueprint for the development of socially responsible AI technologies.

Ultimately, this study aims to bridge the gap between AI efficiency and ethical AI deployment, ensuring that AI-assisted decision-making is not only accurate and scalable but also trustworthy, interpretable, and aligned with human-centric principles. The findings presented in this dissertation pave the way for the adoption of AI technologies that are both innovative and responsible, fostering a future where AI enhances human decision-making while maintaining accountability and fairness.

Chapter 2

Background

2.1 Literature Review: Trustworthy and Efficient Retrieval Pipelines for Multi-Domain Image Data

Early experiments in image retrieval employed hand-engineered features such as SIFT [11], HOG [12], and color histograms [13], which had a tendency to be hard to scale up and generalize across domains. In comparison, modern-day retrieval pipelines leverage deep neural network architectures to learn stable image embeddings end-to-end.

Recent studies have explored combinations of classical and deep learning-based approaches [14, 15], demonstrating improved performance for image retrieval.

2.1.1 Evolution of Deep Learning for Image Retrieval

Deep learning methods have enabled the creation of bespoke and off-the-shelf models that can extract subtle semantic features from images. Groundbreaking works based on ResNet and EfficientNet architectures have demonstrated that deep convolutional models can learn high-level representations that support precise similarity measurements. Current approaches use Vision Transformers (ViT), which, with their increased data requirements, offer the ability to learn long-range dependencies and fine-grained details. These advances have paved the way for trustworthy retrieval pipelines that not only offer effective computing but also offer flexibility across different image domains [16].

2.1.2 Retrieval Pipelines and Multi-Domain Accessibility

A proper retrieval pipeline ought to ensure that the learned embeddings are compatible with similarity search infrastructure at scale. Methods such as K-Nearest Neighbors (KNN) and advanced tools such as Facebook AI Similarity Search (FAISS) are now the standard for ranking image similarity in real-time. In multi-domain scenarios, such as healthcare, agriculture, and general classification tasks, the retrieval system should be able to support various image features. This necessitates a multi-domain paradigm where known architectures and custom-built models are used to match the pipeline with the requirements of specific applications [17].

2.1.3 Fast Computing and Trustworthiness

Such reliable retrieval systems not only need to be precise but also have to be fast and reliable. Fast processing is achieved by using optimized deep neural networks and scalable frameworks that can handle big data streams effectively. Model quantization, parallel processing, and low-latency indexing in the embedding space are crucial techniques that enable low levels of latency for real-time use. Trustworthiness is also increased by making the system's decisions interpretable through interpretable embedding spaces and clear evaluation metrics such as Mean Average Precision (mAP) [18].

2.1.4 Ethical Considerations and Fairness

Since retrieval systems are distributed across various domains, ethical concerns become dominant. Training data bias, embedding representation transparency, and retrieval result fairness are paramount challenges highlighted in recent research. Diversity in datasets across multiple domains guarantees generalization as well as minimizes potential biases that may affect minority groups or less represented applications. Scientists increasingly now call for robust evaluation methodologies and ethical principles to guide the development of retrieval systems towards justice and accountability [19].

2.1.5 Research Gaps

Despite significant progress, a few issues remain. The recent literature shows that while a vast majority of models work very well on benchmark datasets, they struggle in cross-domain scenarios due to the semantic gap and domain-specific artifacts [20, 21].

2.2 History of Image Embeddings

Image embeddings are low-dimensional approximations of images in a high-dimensional vector space. They are developed to be descriptive embedded representations that extract important aspects of an image while reducing the dimension of the image. Fast image analytics, image matching, and other machine learning applications are the key uses of image embeddings. One of the critical properties is that two similar images have embeddings closer to each other in vector space, and two different images are distant from one another. Because of this property, embeddings become central to image recognition, retrieval, clustering, and similarity estimation problems. Over time, image embedding research has gone from hand-designed features to learning representations via deep learning [22, 23].

Before the advent of deep learning, image analysis relied heavily on handcrafted features designed to capture specific aspects of images. Two foundational techniques included:

SIFT (Scale-Invariant Feature Transform)

Introduced by David Lowe in 1999, SIFT extracts distinctive key points from images that are invariant to scaling, rotation, and translation. It identifies salient regions and describes their local gradients, making it effective for object recognition and image matching [24].

HOG (Histogram of Oriented Gradients)

Proposed by Dalal and Triggs in 2005, HOG computes histograms of intensity gradients within localized regions. This method is particularly suited for capturing shape and edge-based features, which is useful for object detection, such as pedestrian detection in surveillance footage [12].

The advent of deep learning in the 2010s brought a paradigm shift in image processing. Although Convolutional Neural Networks (CNNs) were conceptualized in the 1980s, they became practical with increased computational power and large, labelled datasets like ImageNet. CNNs learn hierarchical feature representations directly from raw pixel data, overcoming the limitations of handcrafted features [25].

Several landmark CNN architectures have shaped the evolution of image embeddings:

- **AlexNet (2012):** A pioneering architecture by Krizhevsky *et al.* demonstrated the power of deep networks for image classification. Its victory in the ImageNet competition marked the start of the deep learning era in computer vision [25].
- **VGGNet (2014):** Developed by Simonyan and Zisserman, VGGNet introduced deeper

networks with uniform convolutional filter sizes (3x3). Its simplicity and effectiveness made it a popular choice for feature extraction [26].

- **GoogLeNet (Inception, 2014):** Google researchers introduced the Inception module, which allowed for efficient computation by incorporating multi-scale convolutions in a single layer. This innovation improved both accuracy and computational efficiency [27].
- **ResNet (2015):** ResNet, developed by He *et al.*, addressed the vanishing gradient problem by using residual connections, enabling the training of much deeper networks and significantly boosting performance [28].

One of the most important advances in the history of image embeddings was the application of transfer learning. Transfer learning is a method in which models are pre-trained on large datasets (e.g., ImageNet) and then fine-tuned on smaller, domain-specific datasets for particular tasks. Transfer learning drastically reduces computational requirements and training time while improving performance on specialized tasks. VGGNet, ResNet, and Inception are some of the models that are generally employed as a starting point for transfer learning in applications like medical imaging and agriculture [26, 28, 27].

The transition from hand-crafted features to deep learning representations has transformed computer vision. Breakthrough architectures like AlexNet, VGGNet, and ResNet have enabled effective and robust feature extraction, while transfer learning has rendered these techniques generic across a variety of domains.

2.3 Theoretical Foundations of Image Embeddings

Theoretical underpinnings for image embeddings derive from fundamental concepts in optimization, machine learning, and mathematics. These principles govern the processes of building embeddings, similarity measurement, and optimizing them in a manner to capture beneficial visual patterns. In this section, primary theoretical components like distance measures, learning paradigms, and optimization methods are explored.

Distance Metrics

Distance metrics are essential for comparing image embeddings, as they quantify the similarity or dissimilarity between high-dimensional vectors. Two widely used metrics are:

Euclidean Distance: The Euclidean distance between two embedding vectors x and y is defined as:

$$d(x, y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}, \tag{2.1}$$

where n represents the dimensionality of the vector space.

Cosine Similarity: Cosine similarity measures the cosine of the angle between two different vectors, emphasizing the orientation rather than the magnitude. It is expressed as:

$$\text{Cosine Similarity}(x, y) = \frac{x \cdot y}{\|x\| \, \|y\|}, \tag{2.2}$$

where $x \cdot y$ denotes the dot product, and $\|x\|$ and $\|y\|$ represents the magnitudes of x and y, respectively.

Learning Paradigms

The creation and application of image embeddings rely on various learning paradigms:

Supervised Learning: In supervised learning, models are trained on labeled data. Common loss functions include:

Cross-Entropy Loss (Classification):

$$L = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{C} y_{ij}\log(P_{ij}), \tag{2.3}$$

where y_{ij} is the true label, P_{ij} is the predicted probability, C is the number of classes, and N is the number of samples.

Mean Squared Error (Regression):

$$L = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2, \tag{2.4}$$

where y_i and $\hat{y}_i$ are the true and predicted values, respectively.

Unsupervised Learning: In the absence of labels in datasets, different methods such as clustering (e.g. k-means, hierarchical clustering) and dimensionality reduction (e.g. PCA, t-SNE [29]) help group or simplify data.

Semi-Supervised Learning: This approach leverages both labeled and unlabeled data to improve model performance, particularly in domains where annotated data is limited.

Optimization Techniques

Training models to generate embeddings requires optimization methods that minimize a loss function, thereby refining the embedding space.

Gradient Descent: Gradient descent iteratively updates model parameters to minimize the loss. A common variant, Stochastic Gradient Descent (SGD), updates parameters using random subsets (batches) of data:

$$w \leftarrow w - \eta \cdot \nabla L(w), \tag{2.5}$$

where w represents the model parameters, η is the learning rate, and $\nabla L(w)$ is the gradient of the loss function.

Backpropagation: Backpropagation computes gradients of the loss function with respect to model parameters using the chain rule. This technique, which helped enable the training of deep neural networks [30], makes it possible to adjust weights in the network.

Regularization Techniques: Regularization prevents overfitting by penalizing complex models. Common techniques include:

- **Dropout:** Randomly drops a fraction of network units during training to improve robustness [31].
- **L2 Regularization (Weight Decay):**

$$L_{\text{reg}} = L + \lambda \sum_{i=1}^{n} w_i^2, \tag{2.6}$$

  where λ is the regularization strength.

Dimensionality reduction techniques like t-SNE are normally used to plot high-dimensional embeddings in 2D or 3D. t-SNE embeds, for instance, retain local proximity, and clusters and trends are now easier to grasp. Figure 2.1 illustrates a t-SNE plot.

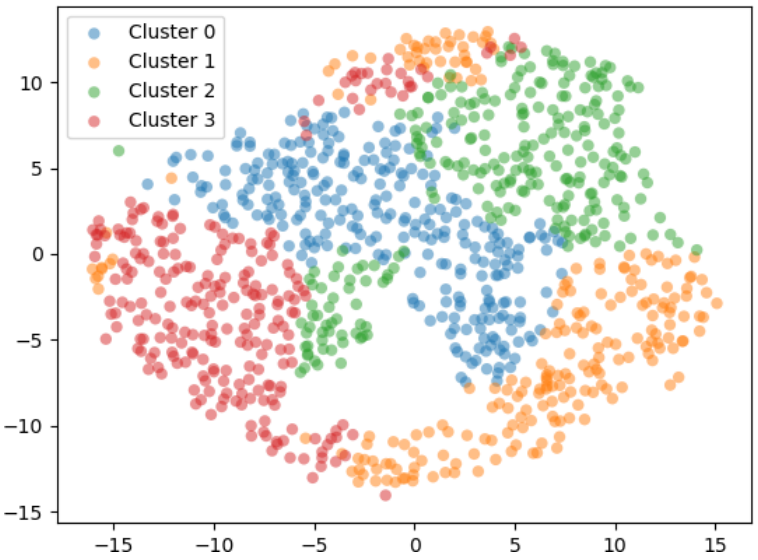

Figure 2.1: t-SNE example for high-dimensional visualization [29].

Theoretical foundations of image embeddings derive from basic distance measure principles, learning paradigms, and optimization techniques. Combining the principles, embeddings allow models to represent and manipulate complex visual data in an efficient way and provide them with application to image retrieval, clustering, and classification.

2.4 Deep Learning Methods for Extracting Image Embeddings

Image embeddings are high-dimensional, dense vectors that hold the content of an image. Deep learning—and specifically Convolutional Neural Networks (CNNs)—has revolutionized the process by enabling automatic, hierarchical feature learning. This section covers the building blocks of CNNs, key architectural innovations, and current best practices for image embedding extraction.

CNNs are among the deepest and most popular image embedding extraction models based on deep learning. They consist of several layers that carry out specific functions in recognizing features from an image:

Convolutional Layers

These layers pre-filter input images to extract local patterns such as edges, texture, and shapes. They facilitate hierarchical learning of feature representations, which is critical for classification and object detection tasks [25].

Pooling Layers

Pooling layers reduce the spatial dimensions of feature maps while preserving essential information. This operation lowers computational complexity and enhances model robustness to small input variations [26].

Fully Connected Layers

At the end of the network, fully connected layers integrate features extracted from previous layers, forming a final vector representation (embedding) of the input image [28].

Over time, CNN architectures have evolved to improve efficiency, scalability, and performance for diverse image embedding extraction tasks. Prominent models include:

- **ResNet50:** Proposed by He *et al.* [28], ResNet50 introduces residual connections to address the vanishing gradient problem, allowing the training of very deep networks. Its embeddings are widely used for image classification and feature extraction.
- **MobileNetV2:** Developed by Sandler *et al.* [32], MobileNetV2 is designed for mobile and embedded applications. Using depthwise separable convolutions and an inverted residual structure, it reduces computational demands while maintaining accuracy.

- **EfficientNetB0:** Proposed by Tan and Le [33], EfficientNetB0 employs a compound scaling method that balances network width, depth, and resolution. This strategy enables state-of-the-art performance with efficient resource usage.

In addition to these CNN-based models, more advanced methods have been developed to further enhance the quality of image embeddings:

Autoencoders

Autoencoders are neural networks that encode input images into a lower-dimensional latent space and then try to reconstruct the original input. Variational autoencoders (VAEs) and convolutional autoencoders are particularly suitable for unsupervised image embedding extraction through learning compact representations of complex data [34].

Siamese Networks

Siamese networks use identical sub-networks to process pairs of images and learn a similarity metric. This approach is effective for tasks like face verification and image retrieval, where differentiating between similar and dissimilar images is critical [35].

Triplet Networks

Generalizing from the Siamese architecture, triplet networks handle three images simultaneously: an anchor image, a positive one, and a negative one. This arrangement warps the embedding space so that similar images are grouped closer to each other and dissimilar images are left more apart [36].

Deep learning image embedding extraction techniques have revolutionized computer vision. Architectures such as ResNet50, MobileNetV2, and EfficientNetB0 offer high-performance, scalable, and efficient methods for extracting features, but in addition, more advanced methods like autoencoders, Siamese networks, and triplet networks also drive increased accuracy and performance in use across different scenarios.

2.5 Application of Image Embeddings

Image embeddings refer to the representation of images as numerical vectors that encode their most important features and patterns. These embeddings enable machines to represent, analyze, and generate visual information—marking a significant milestone for computer vision technologies. This section discusses diversified applications of image embeddings across several domains, including object detection, image segmentation, medical imaging, and agriculture.

Object Detection: Image embeddings represent images as numerical vectors that capture their most important features and patterns. These embeddings enable machines to represent, analyze, and generate visual content—marking a great leap for computer vision technologies. This chapter discusses diversified applications of image embeddings in diverse applications in object detection, image segmentation, medical imaging, and agriculture.

Image Segmentation: Segmentation divides an image into semantically meaningful parts, allowing individual objects or areas to be separated. Image embeddings enhance precision in such applications as the separation of foreground objects from the background and the division of medical images into anatomical structures. For instance, the U-Net architecture [37] has emerged as a favorite for biomedical image segmentation, where it has proved very valuable for surgical planning and monitoring treatment.

Medical Imaging: In the field of medicine, image embeddings are extremely important in diagnosing diseases and planning treatment. Tumors, fractures, and lesions are some of the diseases that can be diagnosed by utilizing embeddings extracted from medical images, i.e., MRI and CT scans. Machine learning models trained on these embeddings can easily detect diseases with high accuracy, which can aid radiologists in making faster and more accurate diagnoses [38]. In addition, longitudinal imaging data-derived embeddings allow for the prediction of disease progression and patient outcomes, making it possible to tailor treatment regimens.

Agriculture: In agriculture, embeddings enable early detection of plant diseases by analyzing high-resolution imagery. They facilitate crop health monitoring and support precision agriculture practices, allowing farmers to optimize yield and reduce losses caused by pests or diseases [39].

Other Applications: Image embeddings are also used in end-to-end learning for self-driving cars, where they contribute to scene understanding and decision-making [40].

Through these diverse applications, image embeddings have transformed fields ranging from healthcare to agriculture, driving significant advances in computer vision.

2.6 Data Collection and Preprocessing

Data collection and data preprocessing are two of the greatest steps in the development of a deep learning model—particularly while creating image embeddings. Adequate data handling facilitates that clean, consistent, well-formatted input enters the model, which is imperative for learning efficiently, developing strong performance, and making proper predictions. Data collection, data preprocessing, and data augmentation have some essential concepts discussed here for an overview of their significance and usage.

Data Sources

The foundation of a successful model lies in the quality and consistency of its data. Training images are often sourced from:

- **Public Datasets:** Popular datasets such as ImageNet [41] and COCO [42] provide large, annotated collections.
- **Custom Sources:** Data can also be collected via web scraping, camera systems, or manual acquisition.

To ensure efficient processing, images should be standardized in file formats (e.g., JPEG, PNG, TIFF) and resolutions. Consistency in formatting minimizes preprocessing overhead and reduces potential errors during pipeline execution.

Preprocessing Techniques

Preprocessing converts raw image data into a format suitable for training, ensuring that the model learns effectively from the data. Key techniques include:

Normalization and Standardization:

- **Normalization:** Scales pixel values to a specified range (typically [0, 1] or [-1, 1]), which helps models converge faster by reducing the impact of varying pixel intensities.

- **Standardization:** Centers and scales image features to zero mean and unit variance. In mathematical terms, for an image with pixel values x, standardization is given by:

$$x_{\text{std}} = \frac{x - \mu}{\sigma}, \tag{2.7}$$

  where μ and σ are the mean and standard deviation of the pixel values.

Contrast Enhancement: Techniques such as histogram equalization adjust image contrast to make essential features more distinguishable.

Handling Class Imbalances: Addressing imbalanced datasets is crucial for unbiased learning. Common methods include:

- **Oversampling/Undersampling:** Balancing class distributions by duplicating minority samples or reducing majority samples.
- **Class-Weighted Loss Functions:** Assigning higher weights to underrepresented classes during training [43].

Data Augmentation

Data augmentation artificially expands the dataset by introducing variations that improve the model's robustness and generalization ability. Key augmentation techniques include:

Geometric Transformations:

- Rotation, translation, scaling, and flipping help the model learn invariance to orientation, size, and position changes.

Colour Adjustments:

- Brightness, contrast, and jittering adjustments simulate varying lighting conditions.

Advanced Techniques:

- **Cutout:** This method randomly masks out regions of an image, preventing over-reliance on specific features [44].
- **Mixup:** Combines images to create synthetic samples, thus improving generalization [45].

Dataset Splitting and Pipeline Design

For unbiased evaluation and model tuning, datasets are typically divided into:

- **Training Set (70-80%):** Used for model learning.
- **Validation Set (10-15%):** Used during training to tune hyperparameters and evaluate intermediate performance.
- **Test Set (10-15%):** Reserved for final performance evaluation.

Maintaining consistent class distributions across these splits minimizes bias and ensures reliable evaluation. In cases of imbalanced datasets, techniques such as stratified sampling can help preserve class proportionality.

A well-designed data pipeline reduces preprocessing and augmentation while utilizing all available resources. Libraries like TensorFlow [46], PyTorch [47], and OpenCV [48] make it easy to build such pipelines by leveraging parallel computing on the GPU and CPU. The salient pipeline features are:

- **Automation:** Consistent handling of preprocessing and augmentation steps for large datasets.
- **Scalability:** Adaptation to various hardware configurations, including GPU acceleration.
- **Error Handling:** Management of issues such as missing or corrupted files and NaN values.

Data processing is more than preparation—robust data processing is the basis of high-quality embeddings and stable model performance. A pipeline that cleans, normalizes, and augments data enables models to generalize better and also assists in fighting overfitting such that the embeddings derived capture semantic information that's helpful for downstream tasks.

2.7 Evaluation Metrics for Image Embeddings

The evaluation of image embeddings is required to ascertain the quality of the features they embody and their suitability for downstream tasks such as classification, retrieval, and clustering. In this section, some of the metrics are presented used to explore the efficacy of image embeddings.

Classification Metrics

General metrics used to assess models built on top of image embeddings include:

Classification Metrics

General metrics used to assess models built on top of image embeddings include:

- **Accuracy:** The proportion of correctly classified instances out of all predictions.
- **Accuracy@10:** The proportion of cases with atleast one correct label in the top 10 retrieved results. This metric is especially useful in retrieval-based applications where users often inspect only the first few results [49].
- **Precision:** The ratio of true positive predictions to all predicted positives, indicating the relevance of the retrieved results.
- **Recall:** The proportion of true positives identified out of all actual positives, reflecting the model's ability to capture all relevant instances.
- **F1-Score:** The harmonic mean of precision and recall, especially useful for imbalanced datasets.
- **Confusion Matrix:** Provides a detailed breakdown of true positives, true negatives, false positives, and false negatives, offering insight into specific misclassification patterns.

The inclusion of Accuracy@10 is particularly motivated by its practical significance in search and recommendation systems. Many retrieval-based applications, such as image search engines, prioritize ranking quality over absolute classification accuracy. Users typically engage only with the top-k results rather than reviewing an entire dataset, making Accuracy@10 and similar ranking metrics more meaningful [50].

Embedding Quality Metrics

Metrics that directly assess the quality and structure of embeddings include:

- **Euclidean Distance:** Measures the straight-line distance between embedding vectors.
- **Cosine Similarity:** Evaluates the cosine of the angle between two vectors, emphasizing directional similarity.
- **Manhattan Distance:** Sums the absolute differences of their coordinates, providing an alternative distance measure.
- **Triplet Loss:** A training metric ensuring that embeddings of the same class are closer together while those of different classes are farther apart.

Retrieval Metrics

For retrieval tasks, the following metrics assess the system's ability to return relevant items:

- **Mean Average Precision (mAP):** Aggregates precision values over multiple queries to yield an overall measure of retrieval quality [51].
- **Precision@K (P@K):** Measures the fraction of relevant items within the top K retrieved results.
- **Recall@K (R@K):** Calculates the proportion of all relevant items that appear within the top K retrieved results.

Visualization Metrics

Visualization techniques help interpret and validate the structure of high-dimensional embeddings:

- **Principal Component Analysis (PCA):** Reduces embedding dimensionality by projecting data onto new axes that capture maximum variance, simplifying analysis while preserving structural information [52].
- **t-Distributed Stochastic Neighbor Embedding (t-SNE):** Maps high-dimensional embeddings into a 2D or 3D space, making clusters and separations visually interpretable [29].

While visualization techniques do not replace traditional quantitative metrics, they serve as a tool for gaining intuitive insights into how well the embeddings represent the underlying data structure. In particular, they provide a qualitative perspective that can reveal patterns, clusters, and anomalies that may not be evident through numerical evaluation alone.

Chapter 3

Ethical Considerations

The development of AI systems, particularly in the area of image retrieval, must meet ethical requirements. In this chapter, the most significant ethical considerations are summarized and discussed, including minimizing bias, transparency, societal impact, and system reliability.

3.1 Bias and Fairness

Fairness in the image retrieval system is a fundamental aspect of this research. Biases in the datasets or the algorithms may vitiate the credibility and usefulness of the system [53]. These biases were compensated for through a multi-level approach. Datasets were selectively collated first, to include large and representative sets of instances. Algorithmic biases were systematically unearthed, quantified, and addressed through design and validation exercises [54].

Existing research indicates that text-image retrieval models' AI-generated images introduce additional layers of bias, which have the potential to distort the fairness of results [55]. This calls for examining the interaction between AI-generated and actual images in retrieval pipelines. Further, dataset representation biases have been discovered to intensify geographical and demographic imbalances, necessitating explicit action to measure and offset these imbalances [56].

The research employed a multi-dataset training, testing, and validation so that the images span a variety of real-world conditions. This minimizes the chances of bias caused by specific features inherent in a single dataset or inconsistent grading. Furthermore, any biases that were found were documented and mitigated through repeated refinement, thereby ensuring the system is fair [57].

In addition, fairness was provided by keeping experimental conditions such as hyperparameters and class distributions fixed so that the results reflect the system's performance rather than artifacts of imbalanced testing conditions. Nevertheless, it has been pointed out that test-time biases can still occur due to distributional mismatches between training and real-world distributions, necessitating post-training correction mechanisms [58].

Emphasis on fairness within this project is essential to designing AI systems that are actually fair and unbiased. There is much more work that needs to be accomplished in formulating best practices that integrate fairness-aware approaches into AI-based retrieval models.

3.2 Transparency and Interpretability

Transparency and explainability of the image search system form the basis of its embedding strategy. As there is increasing complexity in AI systems, the decision-making process gets blackboxed, causing accountability and ethics in alignment concerns. The project aims to address such issues by embedding mechanisms that reveal how decisions are made.

The system indexes and processes image data in high-dimensional embeddings, not just enabling correct retrieval but also allowing for explainable explanation of patterns and relationships. All the system's decisions are made based on interpretable features in these embeddings so that users can monitor and understand how data is processed and how results are delivered [59, 60].

The interpretability problem is most relevant in deep learning-based image retrieval systems where convolutional neural networks (CNNs) learn abstract features that may not necessarily be human interpretable. Saliency maps, layer-wise relevance propagation, and class activation maps (CAM) are a few of the visualization methods that can be utilized to enhance the transparency of such models [61]. Such methods allow the stakeholders to visually analyze which part of an image had the largest effect on the retrieval result.

By promoting transparency, the system fosters user trust and meets the ethical requirements of AI. A transparent system provides access to its internal workings, allowing stakeholders to verify that it functions as intended in a wide range of real-world settings. This emphasis on the interpretability is essential for high-stakes use cases, where understanding the justification behind the decisions is as important as the decisions themselves [62].

Also, AI system explainability is important for regulatory compliance as most organizations today demand AI decision systems to provide clear explanations for their results. Current research emphasizes the creation of human-centered AI systems with approaches that assist users in interacting with, questioning, and refining AI-based decisions [63]. This approach not only ensures accountability but also encourages debugging and refining retrieval models.

In the end, the goal is to create a system that is not only effective but also fully open, in that its complex inner mechanisms are accessible to its users and regulators too.

3.3 Evaluation of System Impacts

Embedding-based image retrieval systems hold tremendous potential across a vast number of industries, such as healthcare and autonomous systems. Their deployment, however, must be assessed with high levels of care to ensure their usage remains ethical and reliable.

In the medical context, where the outcome concerns patients directly, accuracy and timeliness are paramount. These systems can enhance diagnostic efficiency and accuracy but medical image misinterpretation can lead to inaccurate diagnoses and loss of confidence between healthcare professionals [64]. To counter these dangers, rigorous validation processes, constant monitoring of performance, and provisions for regular updation are indispensable. Additionally, human control must be present so that it can intervene if the behavior of the system deviates from typical trends, hence safeguarding patients' safety [65].

Current research highlights that AI-based healthcare systems require transparent validation procedures and regulatory governance for reliability and fairness [66]. Haptic feedback in medical imaging from AI has been viewed as a measure to advance interpretability of diagnosis results and improve surgeon performance in robotic-assisted surgeries [67]. All these innovations go to highlight the importance of systematic assessment before application in clinical environments.

Similarly, in autonomous systems—such as robot farming—accuracy and reliability are of the essence. A failure in image retrieval can contaminate crop health or jeopardize operator safety. To address these requirements, the system design incorporates fail-safe mechanisms that allow for human intervention in case of anomalies being detected. This dual-track approach through technological protectionism with people's supervision will provide secure and dependable operations for real-world applications [68].

AI applications in robotics and precision farming also have challenges in decision reliability when there are environmental changes [69]. Researchers advocate for explainable AI (XAI) methods in such fields, where autonomous systems need to justify their retrieval decisions to human operators [70]. This falls under the overall trend of making AI-based automation more transparent and accountable across industries.

Beyond healthcare and agriculture, embedding-based retrieval systems can affect numerous other domains, each of which has its own ethical issues. Careful attention to system effects is required to ensure that technological advancement does not come at the cost of ethical or societal well-being.

3.4 Human Values and Ethics

The ethical design of AI systems is an important consideration in modern artificial intelligence development. As AI technology becomes increasingly integrated into various different industries across the globe, it is crucial to ensure that these systems align with fundamental human values such as fairness, transparency, and accountability [71] to ensure safe operation and usage. This section explores how embedding-based image retrieval systems can be designed in an ethically responsible manner, adhering to principles of justice, equality, and sustainability.

One of the core challenges in ethical AI development is fairness. Machine learning models must be designed to minimize bias and prevent discriminatory outcomes [72]. To achieve this, datasets must be diverse and representative of different demographic groups to avoid underrepresentation. Additionally, fairness-aware learning techniques should be incorporated to ensure that model predictions do not favor specific populations. Regular audits of AI predictions are necessary to detect and correct any unfair patterns, ensuring a more equitable AI system [73].

Transparency is another fundamental requirement for building trust in AI systems. Many deep learning models operate as "black boxes," meaning that their decision-making processes are difficult to interpret [74]. To address this, different explainability techniques such as SHAP, Shapley Additive Explanations, and LIME, Local Interpretable Model-Agnostic Explanations, can be used to provide users with clear justifications and indications for image retrieval results. Ethical auditing tools further ensure that AI-generated decisions can be reviewed and understood by stakeholders, while open-source documentation offers insights into how the system functions, allowing developers and policymakers to assess its fairness and reliability [75].

Privacy protection is another critical ethical consideration. Since AI systems frequently process sensitive personal data, they must be designed with stringent privacy safeguards. Techniques such as differential privacy help prevent unauthorized access to individual data, while secure federated learning enables models to be trained on decentralized datasets without exposing sensitive information. Moreover, strict data governance policies are necessary to ensure compliance with regulatory standards like GDPR and other global data protection laws [76].

The environmental impact of AI systems has also emerged as a key ethical concern. Large-scale deep learning models require significant computational resources, leading to high energy consumption and increased carbon emissions. To mitigate these effects, AI systems should adopt energy-efficient architectures and optimize resource allocation to reduce unnecessary computations. Additionally, carbon footprint tracking mechanisms can be integrated to monitor and minimize the environmental impact of AI development, aligning technological advancements with sustainability goals [77].

Beyond these technical considerations, AI systems must also be socially responsible. This involves engaging stakeholders in AI governance, ensuring that affected communities have a voice in decision-making processes. Ethical impact assessments should be conducted to evaluate the potential consequences of AI deployment, particularly in high-stakes areas such as healthcare and criminal justice. Regulatory compliance with global AI ethics frameworks, such as those established by UNESCO and the European Union, is also essential to maintaining accountability in AI development [78].

By incorporating the principles of fairness, transparency, privacy, sustainability and accountability. AI-driven image retrieval systems can establish a new ethical standard for responsible AI development. Future research should focus on continuous ethical evaluations to ensure that AI remains aligned with evolving societal norms and values, thereby fostering the development of AI technologies that are both innovative and ethically sound [79].

3.5 Validity and Reliability

Ensuring the validity and reliability of an embedding-based image retrieval system is fundamental to its trustworthiness, effectiveness, and usability. Validity refers to the system's ability to retrieve relevant and accurate images, ensuring that results align with real-world expectations and domain knowledge [80]. Reliability, on the other hand, focuses on the system's ability to consistently reproduce results under various conditions and across different datasets [81]. This section outlines the methods employed in this project to maximize validity and reliability, ensuring that the system meets rigorous standards of robustness and accuracy in real-world applications.

The validity of the system is measured by its precision and recall, which assess its ability to correctly retrieve relevant images while minimizing false positives [82]. To ensure validity, the system undergoes extensive training and validation on diverse, high-quality datasets that encompass multiple real-world variations, including different lighting conditions, resolutions, and occlusions. A multi-stage evaluation process incorporates cross-validation and real-world testing, ensuring that the model generalizes effectively to unseen data [83]. Additionally, user feedback is actively integrated into system refinement to ensure that retrieval outputs align with human expectations and professional standards [84].

Reliability is concerned with the system's ability to deliver consistent results under different conditions. This is achieved through failure detection and mitigation strategies, including failover mechanisms and redundancy protocols that maintain system performance even if certain components fail [85]. Continuous model adaptation through online learning and periodic updates ensures that the system evolves with new data and maintains robustness over time. Furthermore, the system

strictly adheres to AI performance standards, such as the ISO/IEC 25010 framework for software quality evaluation, to meet industry-wide reliability and stability benchmarks [86].

To comply with established AI testing and evaluation standards, this project follows a structured benchmarking methodology. The system is tested against standardized datasets commonly used in academic and industrial AI research, such as ImageNet, COCO, and OpenAI benchmarks. Model performance is continuously assessed using key reliability metrics such as F1-score, robustness index, and confidence intervals [87]. Stress testing and adversarial attack simulations are also conducted to evaluate the system's resilience against potential biases, adversarial manipulations, and domain shifts [88].

Beyond technical reliability, trust in AI systems is essential. Ethical reliability is ensured through transparency in decision-making, allowing users to understand how retrieval results are generated [89]. Bias mitigation strategies are incorporated to prevent the retrieval system from favoring certain demographics, categories, or data distributions unfairly. Additionally, regulatory compliance is maintained to align the system with AI governance frameworks such as the EU AI Act and IEEE AI Ethics Guidelines.

By rigorously addressing both validity and reliability, this project aims to deliver an image retrieval system that users can trust for informed decision-making. The integration of robust training, continuous testing, ethical guidelines, and industry benchmarks ensures that the system is reliable, fair, and adaptable to real-world challenges [90].

Chapter 4

Datasets and Data Considerations

4.1 The Need for Datasets with Variance

Datasets are crucial for benchmarking the performance of deep learning frameworks, especially image retrieval systems. Variability in the dataset ensures that the system is able to generalize across domains and scenarios and therefore enables one to thoroughly test its efficiency.

Testing against datasets from different domains, medical imaging, agriculture, and generic classification (e.g., ImageNet [41] and COCO [42]), allows the system to learn how to be resilient to real-world uncertainties, resulting in lower bias and more dependable outcomes. It is this iterative testing cycle that exposes loopholes, making it possible to make continuous improvements and the system strong and reliable.

Recent research focuses on the importance of variance, preserving deep metric learning in image retrieval based on content, such that models learned from diverse sets outperform models learned from domain-specific sets in achieving better feature generalization [91]. Further, utilizing scale varied and intra-class diversified datasets plays an important role in improving the ability of deep learning models for accurate image retrieval in multiple classes [92].

For instance, deep convolutional feature representations have been found to be more robust when trained on datasets that show a broad spectrum of image variations, including illumination changes, occlusions, and distortions [93]. Similarly, researchers have identified that instance retrieval performance is significantly enhanced when models are trained on datasets that cover a variety of visual domains [94]. This highlights the need for incorporating variance in training data to obtain resilient and scalable deep learning systems.

4.2 Agriculture - Leaf Disease Dataset

The selected Leaf Disease Dataset, Kaggle [95], is comprised of 45 classes of plant leaf disease. It contains images that represent diverse crops, diseases, and symptoms and thus provide rich real-world information required for agricultural uses. This dataset is particularly of interest to our project because it replicates actual agricultural situations in which minor changes in disease-specific

features have considerable effects on the retrieval system's performance.

As can be seen from Figure 4.1, the dataset includes diverse instances that challenge the system to focus on disease-specific features. This dataset would facilitate training the system to identify diseases with high precision, thereby enabling farmers to initiate appropriate interventions—possibly reducing the use of pesticides, improving the yield of crops, and aiding in sustainable farming. Despite some limitations regarding class balance and quality of images, the dataset provides an excellent base for conducting agricultural AI research.

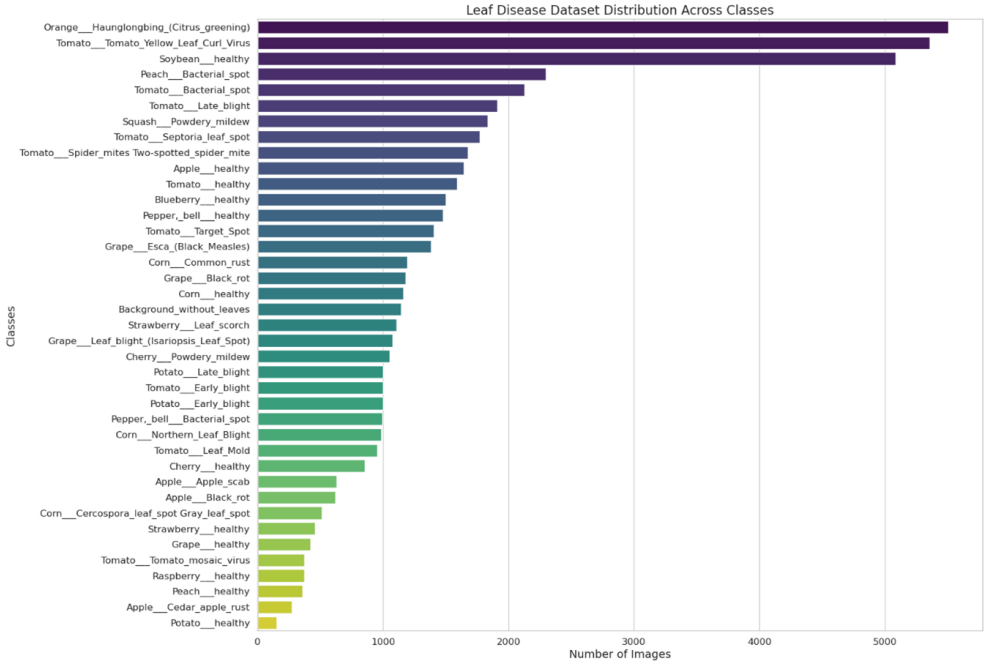

Figure 4.1: Sample images from the Leaf Disease Dataset.

4.3 Healthcare MRI - Brain Tumour MRI Dataset

The selected Brain Tumour MRI Dataset [96] includes over 3,000 brain image MRI scans used in the classification of tumours into 44 groups, including benign, common, and very rare. These images are of high resolution and take account of small details required to identify tumours correctly.

The dataset is important for the project because it acts as a testbed for assessing the system's precision in retrieving and classifying medical images. In medicine, where diagnostic choices have a profound influence on patient outcomes, very high precision is very important. This dataset allows the system to learn the inherent characteristics of different types of tumours, enabling proper retrieval and classification. Inclusion of this dataset encourages the broader mission of developing trustworthy AI systems for healthcare, with reliability and interpretability being top priority. The

dataset provides anonymized data, and ethical standards are maintained as automated diagnostic methods are advanced.

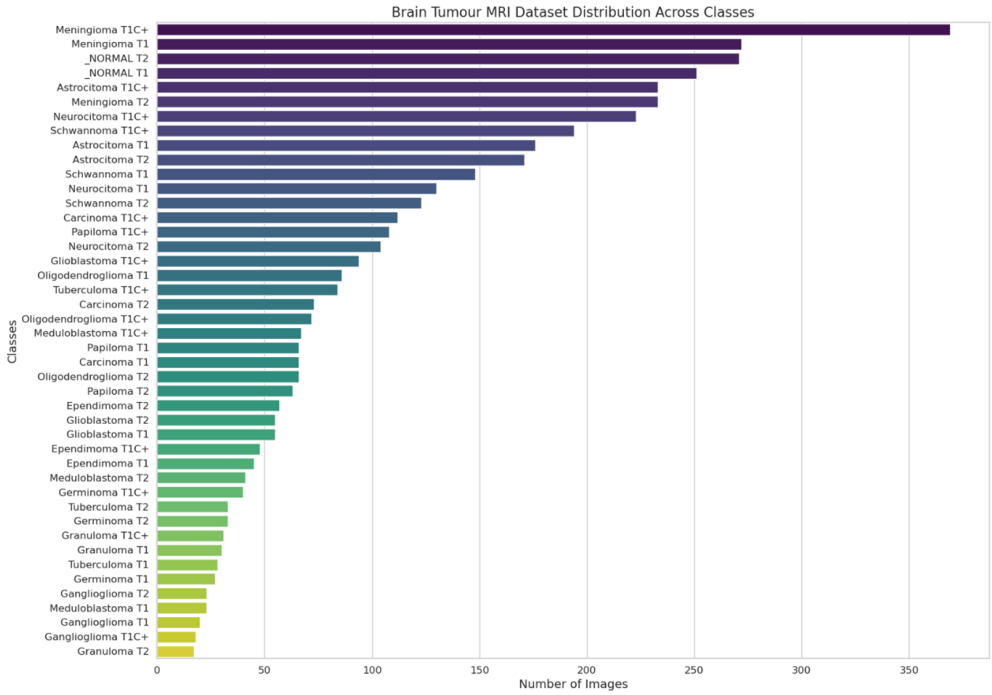

Figure 4.2: Sample MRI images from the Brain Tumour MRI Dataset.

4.4 Healthcare X-Ray – COVID-19 Dataset

The selected COVID-19 Digital X-rays Forgery Dataset [97] consists of close to 15,780 images spread over nine classes, which include real and fake X-rays of COVID-19-infected lungs, viral pneumonia, and other pathologies. It has examples of digital forgery techniques, such as copy-move and splicing, that simulate cases of manipulated medical imaging.

This dataset is very helpful in training the systems to distinguish between original and falsified medical images. Fake images are very risky in medical diagnosis since confusion may lead to the use of the wrong treatments. This project utilizes this dataset to enhance the anomaly detection capacity of the system to ensure the credibility of medical image analysis. Despite the heterogeneous nature of the dataset, it emphasizes the need for developing retrieval systems that are resilient in challenging conditions. This study contributes to the advancement of digital forensic capabilities in medicine and ensures the integrity of medical imaging.

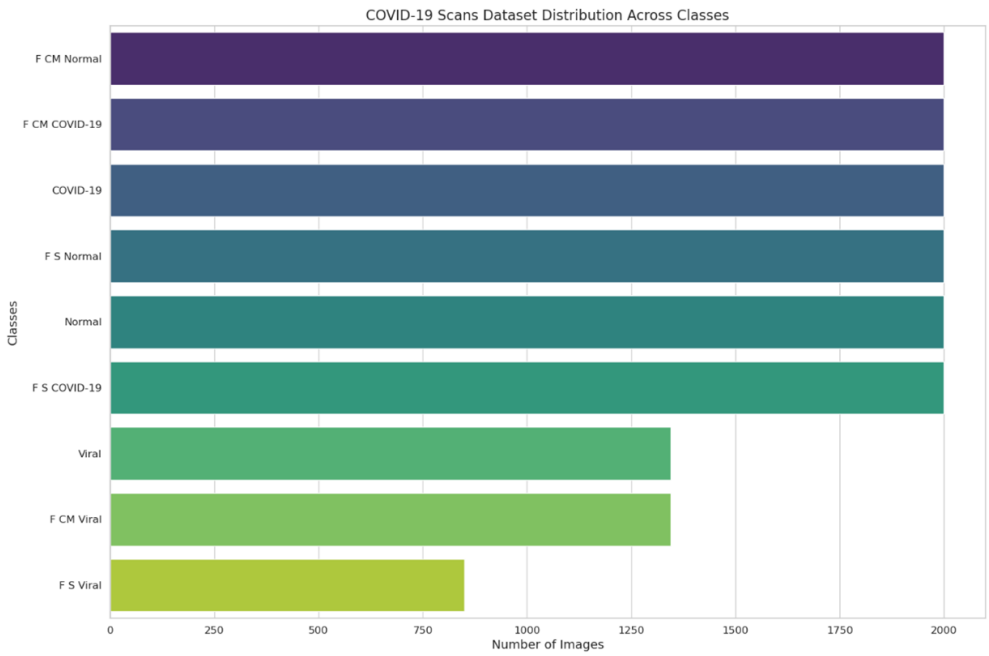

Figure 4.3: Sample images from the COVID-19 Digital X-rays Forgery Dataset.

4.5 General Classification – Pokémon Dataset

The selected Pokémon Dataset [98] is a custom 7,000 image dataset tailored to a lightweight classification problem between 150 diverse Generation One Pokémon. Each photograph has been carefully hand-selected in order to rule out redundant background information, thereby increasing clarity and making it simpler to place primary emphasis on the main subject.

Though this dataset is largely niche, it enables the evaluation of the retrieval system's performance on less straightforward classification instances. The consistent image quality and little background variation enable it to serve well in testing the system's general retrieval capacity at its most fundamental level. But the uniformity of subject matter limits its application in larger domains.

Despite these limitations, the data demonstrates the flexibility of the retrieval system, showing its applicability in numerous fields ranging from specialized domains like medical imaging to general classification problems. Figure 4.4 provides a preliminary overall breakdown of the classes in the dataset, and Figure 4.5 provides the rest of the classes used for classification.

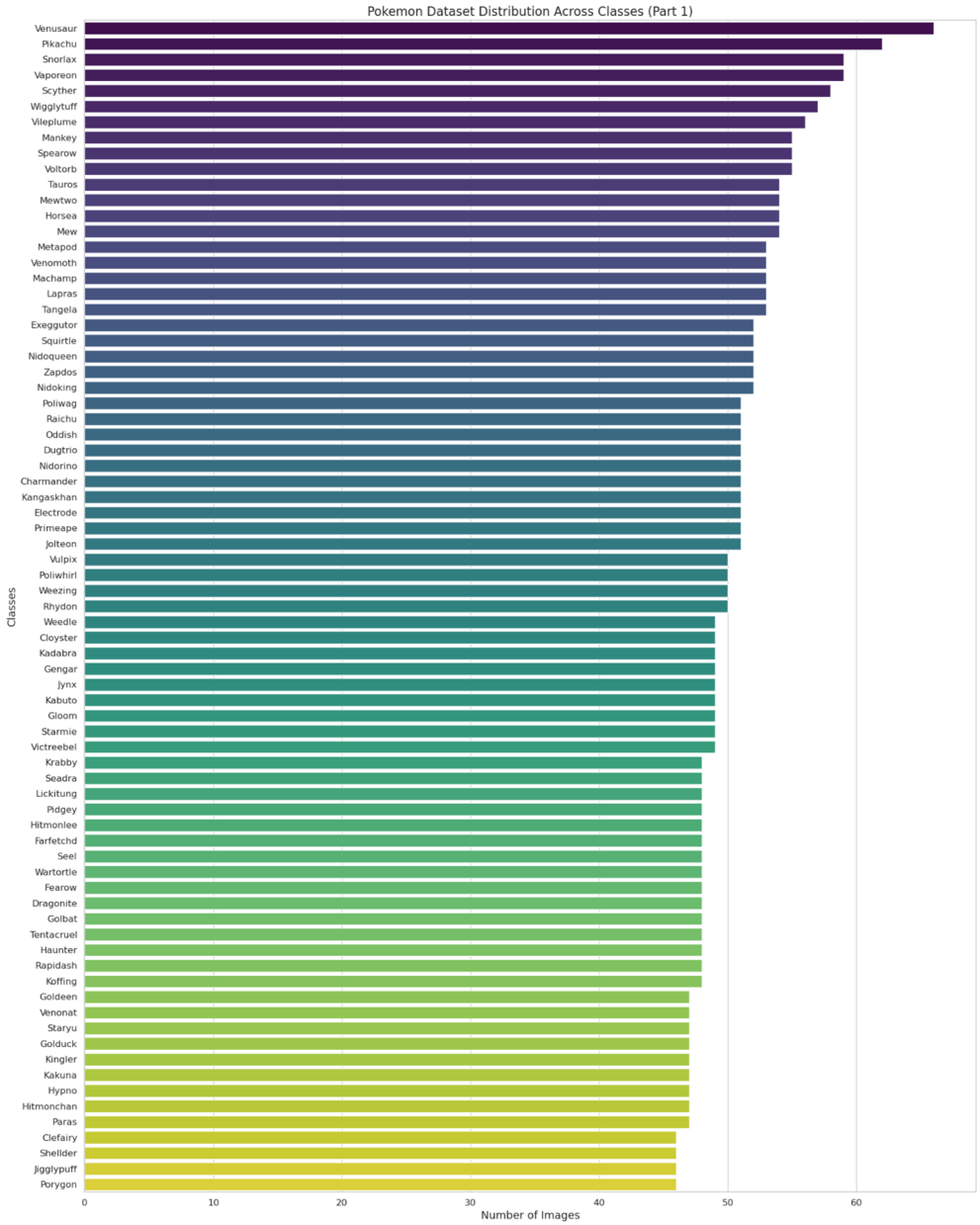

Figure 4.4: Overview of the Pokémon Dataset showing first portion of the 150 Generation One Pokémon classes.

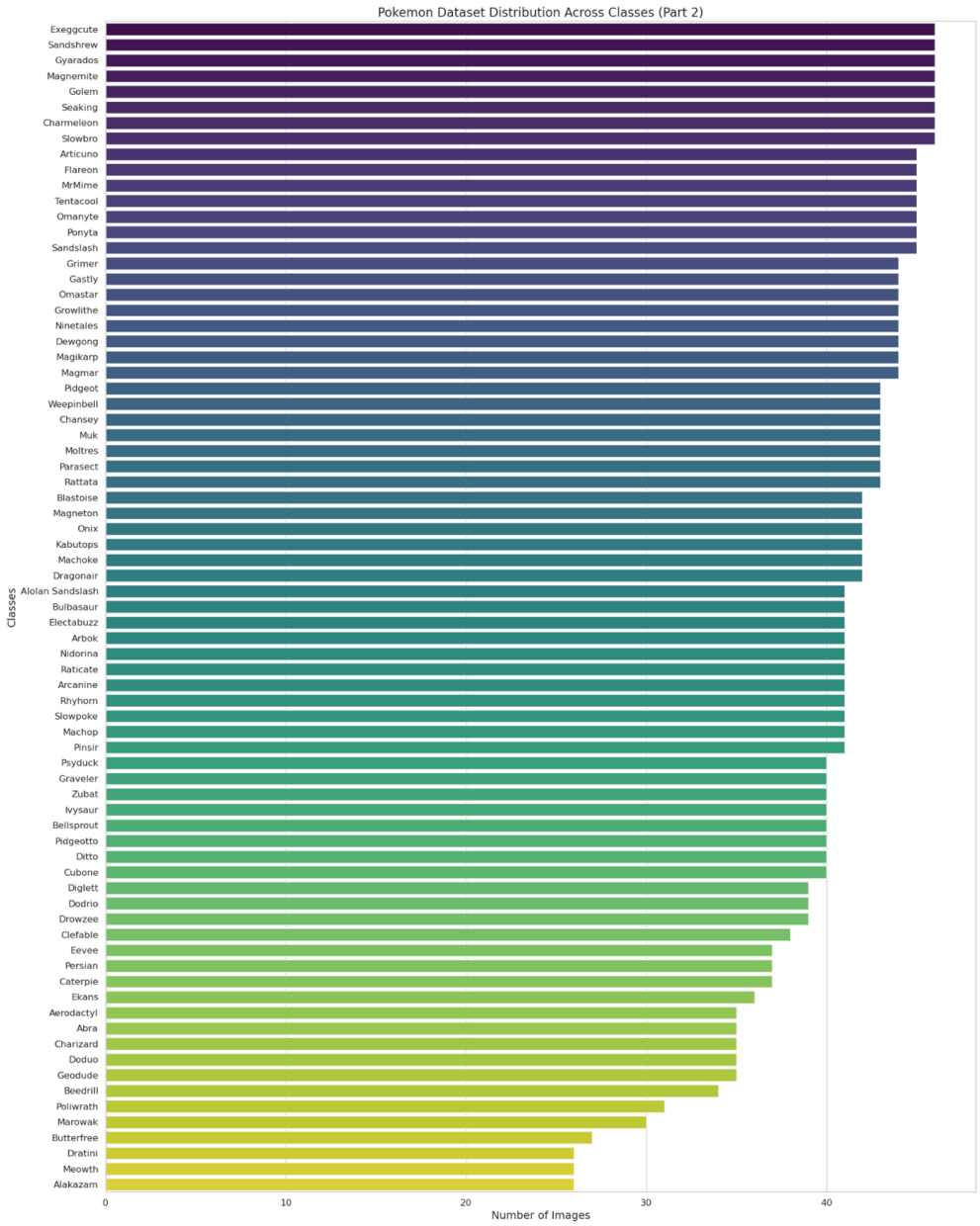

Figure 4.5: Overview of the Pokémon Dataset showing second portion of the 150 Generation One Pokémon classes.

4.6 Visual Comparisons of Datasets

To provide a full representation of the datasets, two main visualizations are presented. The plots depict both the data size and distribution in different datasets and how it helps in understanding their impact on model generalization, training performance, and fairness.

4.6.1 Total Number of Images Across Datasets

The total number of images contained in each of the datasets utilized in this study is indicated in the first plot (Figure 4.6). Dataset size is one crucial factor that influences the robustness and generalizability of deep learning models [99, 100]. Large datasets minimize overfitting and allow deep learning to learn more abstract feature representations, whereas small datasets may lead to biased models with poor real-world generalization.

In addition, dataset balance also affects retrieval performance, particularly for multi-domain image retrieval because an unbalanced dataset would cause model learned representations to be biased towards [101]. For this purpose, in the compilation of this dataset collection, diversity while balancing dataset size across domains was guaranteed and the possibility of model bias was reduced.

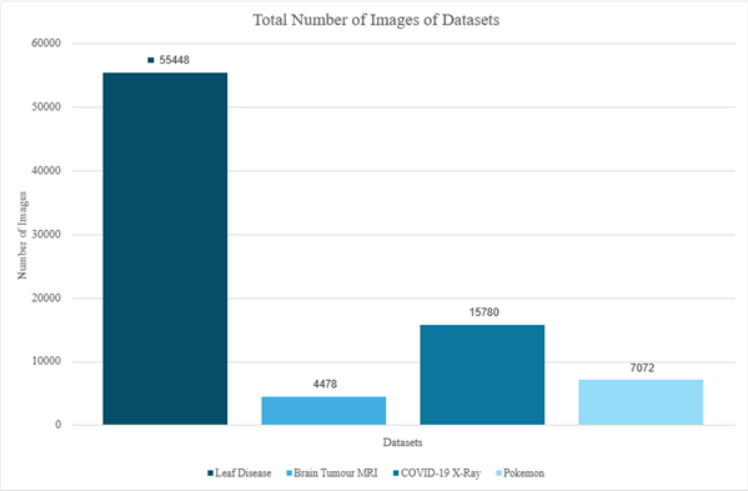

Figure 4.6: Total number of images available across all datasets used in this study. The volume of data is a crucial factor in ensuring robust deep learning models.

4.6.2 Dataset Splitting: Training, Validation, and Testing

The following graph (Figure 4.7) shows how datasets are split into training, validation, and test subsets. Proper dataset split is important in order to estimate model performance on an unbiased scale. A proper split guarantees that:

- The training set is large enough to learn meaningful features without overfitting.
- The validation set provides an unbiased estimate for hyperparameter tuning and model selection.
- The test set remains untouched during model development and serves as the final benchmark.

Recent research shows that typical splits (e.g., 80-10-10 or 70-15-15) may not be optimal in all cases, especially for low-data regimes [102]. Therefore, dataset splitting in this work was done with proper regard to domain constraints, availability of data, and representational fairness [103].

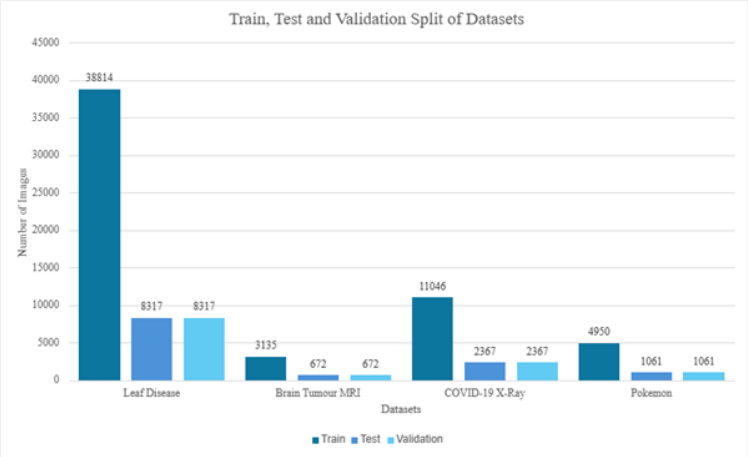

Figure 4.7: Train, test, and validation split of the datasets. Proper dataset partitioning is essential for reliable evaluation and generalization.

4.6.3 Ensuring Generalization Through Diverse Splitting Strategies

The importance of dataset splitting methods has been underscored in domain adaptation and transfer learning research, where test and training distributions typically mismatch [104]. Dataset splits in this work were specifically designed to ensure:

- Representation of all major classes in training, validation, and test sets.
- No data leakage, ensuring that visually similar images do not appear in both training and testing.
- Diverse domain representation, to simulate real-world scenarios.

By taking advantage of knowledge from large-scale dataset training research [105], initial experiments were performed to evaluate the impact of varying split ratios on performance stability, further narrowing down the choice of the final dataset partitions.

4.6.4 Impact of Dataset Composition on Model Fairness

A central aspect of dataset selection is its effect on bias and fairness in image search models. Studies have shown that model bias is exacerbated by imbalanced data distributions—i.e., face recognition datasets with mostly particular communities perform poorly on minority populations [106]. To mitigate this:

- Balanced sampling was employed to ensure no single category dominates.
- Bias auditing methods were applied to the dataset prior to training.

4.7 Dataset Class Imbalances

To quantify the imbalance of image counts per class in our datasets, we computed three statistics: the Imbalance Ratio (IR), the Coefficient of Variation (CV), and the Normalized Entropy. These statistics provide complementary views of the distribution of images per class and tell us whether a dataset is balanced or skewed towards certain classes [107, 108].

Imbalance Ratio (IR)

The Imbalance Ratio (IR) is defined as the ratio between the image count of the most frequent (majority) class and that of the least frequent (minority) class [109]:

$$\text{IR} = \frac{\max\{N_c : c \in C\}}{\min\{N_c : c \in C\}}, \tag{4.1}$$

where N_c is the number of images in class c and C is the set of all classes in the dataset. An IR close to 1 indicates similar representation among classes, whereas a high IR indicates significant imbalance.

Coefficient of Variation (CV)

The Coefficient of Variation (CV) measures the relative variability of class counts. It is calculated as the standard deviation of the class counts divided by their mean:

$$\text{CV} = \frac{\sigma(N)}{\mu(N)}, \tag{4.2}$$

where $\sigma(N)$ is the standard deviation and $\mu(N)$ is the mean of the image counts across classes. A higher CV indicates a greater imbalance [108].

Normalized Entropy

Normalized Entropy is derived from Shannon's entropy, which measures the uncertainty in a probability distribution. For class counts, the entropy H is computed as:

$$H = -\sum_{c \in C} p(c) \log p(c), \tag{4.3}$$

with

$$p(c) = \frac{N_c}{\sum_{c \in C} N_c}. \tag{4.4}$$

The normalized entropy is then given by:

$$H_{\text{norm}} = \frac{H}{\log |C|}, \tag{4.5}$$

where $\log |C|$ is the maximum possible entropy for $|C|$ classes. A H_{norm} close to 1 indicates a nearly uniform (balanced) distribution, while lower values indicate imbalance [110].

Summary of Metrics

Table 4.1 summarizes the metrics computed for each dataset:

Table 4.1: Imbalance Metrics Across Datasets

Dataset	Imbalance Ratio (IR)	Coefficient of Variation (CV)	Normalized Entropy
Leaf Disease	36.23	0.88	0.92
Brain Tumour MRI Scans	21.71	0.84	0.92
COVID-19 Scans	2.35	0.25	0.99
Pokémon	2.54	0.15	1.00

Impact of Dataset Imbalance on Learning

Deep learning models trained on imbalanced data learn biased representations, overrepresenting majority classes and underrepresenting minority classes [111]. Recent work shows that integrating data augmentation and resampling techniques (e.g., SMOTE, evolutionary optimization) can solve these issues by learning more balanced [110, 112].

Dimensionality Reduction Analysis

In order to analyze how dataset characteristics influence the learned feature space, we performed dimensionality reduction on the image embeddings.

Principal Component Analysis (PCA) was first applied to reduce the embedding dimensions to 50, providing a balance between preserving meaningful variance and removing noise from high-dimensional representations. This intermediate reduction helps retain the most significant structure while filtering out less informative components, which can otherwise lead to overfitting or distort visual patterns in subsequent steps.

After PCA, t-Distributed Stochastic Neighbor Embedding (t-SNE) was used to project the embeddings into a two-dimensional space. This allowed a visual perspective to observe the clustering tendency of images across different classes and dataset splits (train, test, and validation) while making the visualization computationally more efficient and interpretable.

The following figures present the t-SNE visualizations for each dataset:

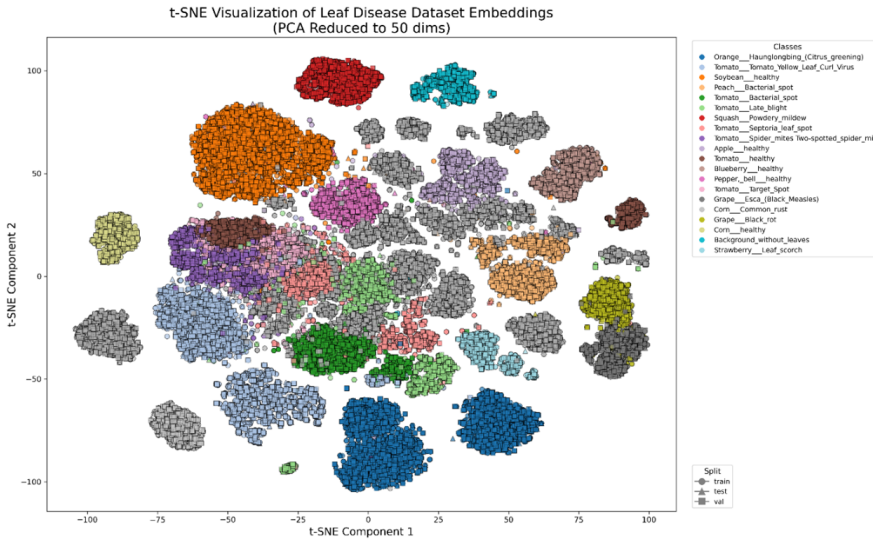

Figure 4.8: t-SNE Visualization of Leaf Disease Dataset Embeddings (PCA 50 dims). Large, dense clusters correspond to majority classes, while minority classes appear in smaller pockets (IR = 36.23).

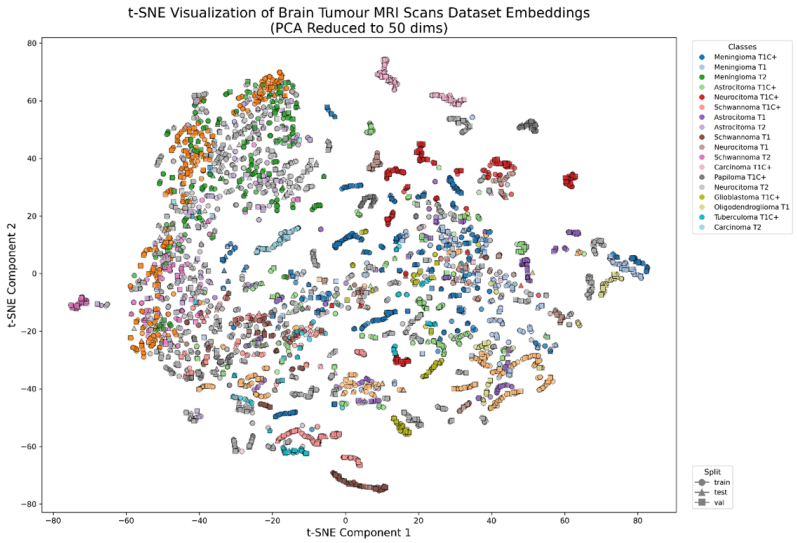

Figure 4.9: t-SNE Visualization of Brain Tumour MRI Scans Dataset Embeddings (PCA 50 dims). Majority classes form dense clusters, whereas minority classes are more dispersed (IR = 21.71).

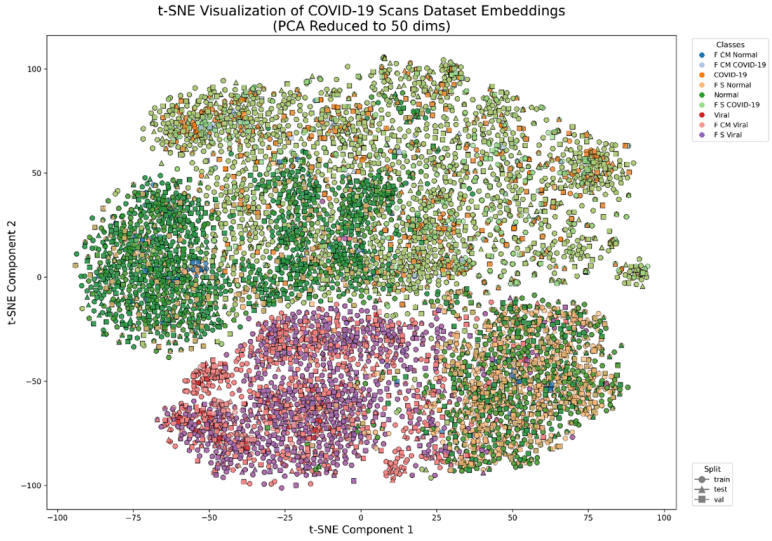

Figure 4.10: t-SNE Visualization of COVID-19 Scans Dataset Embeddings (PCA 50 dims). Classes are nearly uniformly distributed (IR = 2.35).

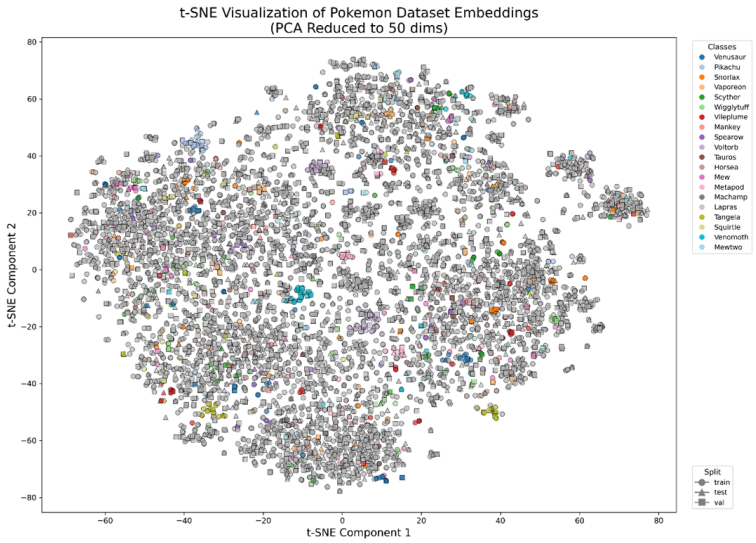

Figure 4.11: t-SNE Visualization of Pokémon Dataset Embeddings (PCA 50 dims). The near-uniform class distribution is evident (IR = 2.54).

The visualization above in Figure 4.11 highlights that only a subset of the 150 possible classes is actively color-coded, while the majority of the dataset remains in grayscale. This is due to the Pokémon dataset containing a large amount of classes. The large presence of grey points indicates the broader dataset distribution, where many classes are not explicitly distinguished in this visualization. Despite this, meaningful clustering patterns, demonstrating the ability of the learned embeddings to separate different categories can still be observed on the subset of classes.

Dataset imbalance has a direct impact on feature learning, clustering behavior, and model generalization. Monitoring dataset balance using Imbalance Ratio (IR), Coefficient of Variation (CV), and Normalized Entropy is crucial, along with applying techniques like SMOTE and adaptive resampling to offset bias [111, 112]. Addressing dataset imbalance is essential for constructing robust and unbiased image retrieval models.

4.8 Background Inclusion: Insights and Implications

Backgrounds of image sets play an important role in impacting the performance of computer vision. Experiments showed that background difficulty has a significant influence on the accuracy of the deep learning model, even causing biases and unexpected relationships [113, 114]. Each data set has unique problems:

- **Agriculture:** Images often include complex backgrounds, such as shadows or natural textures, which can obscure disease-specific features. Effective segmentation techniques are essential to isolate these features [114].
- **Medical Imaging:** MRI scans typically feature controlled and minimal backgrounds that focus solely on anatomical structures. However, variations in patient positioning or imaging conditions can introduce inconsistencies that models must learn to navigate [115].
- **Forgery Detection:** In X-ray images, distinguishing genuine anomalies from forged elements requires algorithms capable of understanding subtle contextual features. Studies suggest that background noise filtering can significantly improve model accuracy in these domains [116].
- **General Classification:** The Pokémon dataset generally presents simple, uncluttered backgrounds that minimize distractions. However, inconsistencies in image quality may still affect model generalization. Research on image distortions has highlighted how even minor variations in backgrounds can degrade deep learning performance [117].

Current work reveals that background suppression and augmentation methods can be employed to reverse such effects [118]. For example, saliency-based object detection methods have been proposed to ensure models focus on the object of concern and not the background, thereby improving robustness [119].

By addressing these challenges, the project ensures that the retrieval system operates effectively across diverse real-world conditions, thereby enhancing its robustness and applicability.

4.9 Exploring Deep Learning Techniques for Background Removal

Background removal is an essential pre-processing technique in computer vision that forms the backbone for many object-focused applications, especially when images have complicated or noisy backgrounds. Removing extraneous visual information can improve classification accuracy by

allowing models to focus on the relevant foreground. In this project, the *Rembg* algorithm, a state of the art module powered by neural networks is employed to remove backgrounds. Its effectiveness, however, depends on the specific characteristics of each dataset.

4.9.1 Rembg Algorithm

The *Rembg* algorithm specializes in segmenting the foreground from the background using advanced neural network architectures. This process is especially useful for datasets with distracting or noisy backgrounds, enabling the model to focus on the pertinent features.

Agriculture – Leaf Disease Dataset Findings: Application of the *Rembg* algorithm on the Leaf Disease Dataset yielded promising results. Noisy backgrounds with shadows, were effectively removed to allow the model to focus on diseased leaves. Key findings include:

- **SSIM Score:** Processed images achieved an SSIM score of 0.4093, indicating moderate similarity between the original and processed images.
- **Impact:** Background removal enhanced interpretability and reduced noise.

Figure 4.12: Rembg algorithm results on a Leaf Disease sample image.

Healthcare MRI – Brain Tumour MRI Dataset Findings: For the Brain Tumour MRI Dataset, the *Rembg* algorithm is less applicable since MRI images typically exhibit uniform, minimal backgrounds (the brain is in sharp contrast to a dark background), rendering background removal redundant.

When using the *Rembg* algorithm on an MRI scan image, the loss of color likely occurs due to the way the tool processes foreground and background separation. Many MRI scans use false-color

maps, such as heatmaps, to enhance visualization, but *Rembg* may strip this information, reverting the image to grayscale.

Additionally, the tool often introduces transparency, and some image viewers interpret transparent regions as black and white. If the MRI scan had a non-standard color profile or embedded metadata, *Rembg* might not retain it, leading to a default grayscale rendering.

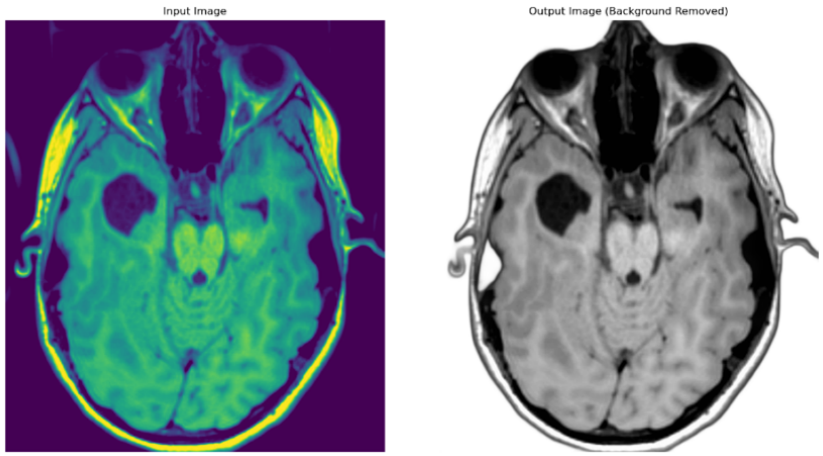

Figure 4.13: Rembg algorithm results on a Brain Tumour MRI sample image.

Healthcare X-Ray – COVID-19 Digital X-rays Forgery Dataset Findings: Similarly, the COVID-19 Digital X-rays Forgery Dataset does not significantly benefit from background removal. X-ray images naturally lack distracting backgrounds, as they focus on anatomical structures.

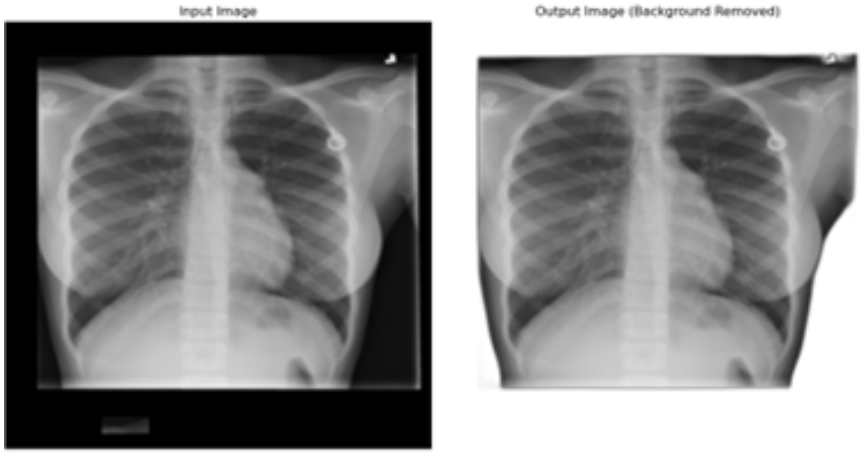

Figure 4.14: Rembg algorithm results on a COVID-19 X-ray sample image.

General Classification – Pokémon Dataset Findings: In the Pokémon Dataset, the *Rembg* algorithm proved useful by removing varied backgrounds, thus allowing the model to focus on the

characters. Key findings include:

- **SSIM Score:** Processed images achieved a higher SSIM score of 0.5493, indicating closer similarity between original and processed images compared to the Leaf Disease Dataset.
- **Impact:** Background removal improved the model's generalization and classification accuracy.

Figure 4.15: Rembg algorithm results on a Pokémon sample image.

4.9.2 Efficacy and Implications of Background Removal

The effectiveness of background removal varies significantly across datasets. One of the key metrics used to evaluate the visual similarity between an original and a processed image is the Structural Similarity Index (SSIM). SSIM compares luminance, contrast, and structural information between two images. It is typically computed as:

$$\text{SSIM}(x, y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)}, \tag{4.6}$$

where μ_x and μ_y are the means, σ_x^2 and σ_y^2 are the variances, σ_{xy} is the covariance of images x and y, and C_1 and C_2 are small constants to stabilize the division [120].

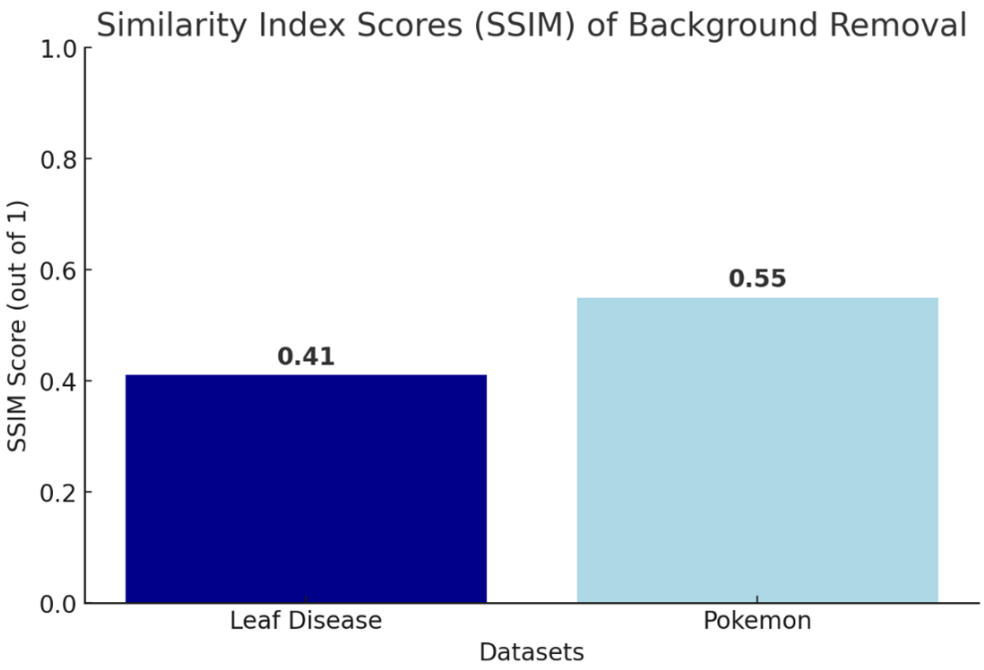

Figure 4.16: Similarity Index Scores (SSIM) of background removal on selected datasets.

These findings underscore that while background removal can be a powerful pre-processing tool, its utility depends on the dataset:

- **Effective Use Cases:** Datasets with noisy or varied backgrounds (e.g., Leaf Disease and Pokémon) benefit substantially from background removal.
- **Limited Utility:** Datasets with uniform or minimal backgrounds (e.g., Brain Tumour MRI and COVID-19 X-rays) show little improvement.

Overall, the evaluation of the *Rembg* algorithm demonstrates the importance of tailoring pre-processing techniques to the specific characteristics of each dataset to ensure optimal performance.

4.9.3 Other Background Removal Techniques Explored

Beyond Rembg, several alternative background removal techniques were investigated to assess their suitability for our application. The evaluation focused on four critical factors:

- **Segmentation Accuracy:** How well the model separates the foreground from the background while preserving fine details.
- **Computational Efficiency:** The memory and processing power required to run inference, especially for large-scale datasets.

- **Ease of Integration:** The complexity of implementing the method within an existing image retrieval pipeline.
- **Preservation of Critical Foreground Features:** The extent to which important object details are retained after segmentation.

Segment Anything Model (SAM): The Segment Anything Model (SAM) by Kirillov et al. [121] is a foundation model for zero-shot and few-shot segmentation tasks. SAM consists of a transformer-based image encoder with a promptable mask decoder, allowing it to generalize to a wide range of segmentation problems.

Despite its high flexibility and state-of-the-art accuracy on diverse segmentation benchmarks, SAM posed several challenges for our use case:

- **Computational Overhead:** The ViT-based backbone of SAM requires significant GPU resources, making real-time processing impractical in retrieval pipelines [122].
- **Integration Complexity:** Unlike traditional U-Net-based architectures, SAM's dependency on prompt engineering made the integration into the retrieval pipeline complex.
- **Inference Speed:** On a NVIDIA RTX 3090, SAM took 0.5–1.2 seconds per image. this is significantly slower than other models [123]. A benchmark study comparing SAM across 22 consumer GPUs confirmed that it has a low inference FPS (1–2 FPS on RTX 3090) compared to other models tested [122].
- **Unnecessary Generalization:** SAM's biggest strength lies in its ability to segment unknown objects, but for these controlled dataset environments, this generalization / feature is unnecessary.

Thus, while SAM is a really impressive tool, its real-time usability and capability in the proposed retrieval system was limited due to high hardware requirements and complex integration challenges.

U-2-Net: The U-2-Net model, developed by Qin et al. [124], is a lightweight deep learning model optimized for object detection. Its nested U-structure enhances fine-grained segmentation while maintaining a low computational footprint.

Advantages of U-2-Net include:

- **Efficient Feature Extraction:** Its architecture captures multi-scale context, making it particularly effective for detecting fine object boundaries.

- **Lower GPU Requirements:** Unlike SAM, U-2-Net can be deployed on mid-tier GPUs such as an NVIDIA GTX 1660 Ti, running inference at 25–40 FPS [125, 124, 126].

Despite its efficiency, U-2-Net's lower segmentation fidelity made it less effective for high-precision retrieval tasks such as really complex edges and features.

DeepLabV3+: DeepLabV3+ [127] is a state-of-the-art semantic segmentation model that extends DeepLabV3 by incorporating atrous spatial pyramid pooling (ASPP) for improved feature extraction at multiple scales.

Strengths:

- **Robust Semantic Segmentation:** DeepLabV3+ delivers high segmentation accuracy across various medical imaging, object detection, and satellite image processing, etc.
- **Fine-Grained Detail Preservation:** The use of encoder-decoder architecture allows it to maintain fine spatial details.

However, the main limitations observed include:

- **High Computational Cost:** DeepLabV3+ requires significantly more memory, making it impractical for real-time applications [128, 129].
- **Slower Inference Time:** - On an NVIDIA RTX 3090, inference takes 180–260 ms per image [130, 131, 132].
- **Training Overhead:** Fine-tuning DeepLabV3+ for the specific datasets used in this research required extensive hyperparameter tuning, adding to overall complexity.

While DeepLabV3+ provided the best segmentation accuracy between all methods, its computational requirements and inference latency made it unsuitable.

Rationale for Choosing Rembg: After evaluating the alternatives, Rembg was chosen as the best balance between segmentation accuracy and computational efficiency.

Key reasons for this selection:

- Low Computational Overhead: Rembg is optimized for fast processing, making it suitable for real-time applications [133].

- High Efficiency on Structured Backgrounds: It performed well on medical imaging and synthetic datasets where backgrounds followed predictable patterns.
- Ease of Integration: Unlike SAM or DeepLabV3+, Rembg did not require extensive prompt engineering or retraining, simplifying deployment in our retrieval pipeline.
- Consistent Foreground Feature Preservation: In contrast to U-2-Net, Rembg preserved finer object details, making it better suited for retrieval-based applications.

Comparison of Background Removal Techniques

Table 4.2 summarizes the key findings from the analysis.

Table 4.2: Comparison of Background Removal Techniques

Method	**Accuracy**	**Speed (FPS)**	**Hardware Req.**	**Integration Complexity**
SAM	**High**	1–2 FPS	**High (GPU-heavy)**	Complex
U-2-Net	Medium	25–40 FPS	Low	Medium
DeepLabV3+	**High**	4–6 FPS	**High (VRAM-intense)**	Complex
Rembg	**Balanced**	**45+ FPS**	**Low (CPU/GPU-friendly)**	**Easy**

Through analysis, Rembg was the most feasible solution, balancing speed, accuracy, and ease of deployment. While SAM and DeepLabV3+ yielded better segmenting accuracy, their hardware constraints and higher inference times made them unfeasible for real-time image retrieval. U-2-Net, though efficient, was not effective with domain-specific images, making Rembg the best choice for our application.

Chapter 5

Deep Learning for Image Retrieval: Models, Methods, and Pipelines

5.1 Deep Learning Framework Overview

Deep learning frameworks are the fundamental building blocks for designing, training, and deploying intricate neural network structures. They support AI breakthroughs across a wide range of areas by providing robust infrastructures for model design and deployment. This section highlights key aspects of two leading frameworks, TensorFlow and PyTorch, and explains why TensorFlow was chosen for this project's embedding-based image retrieval system.

5.1.1 TensorFlow vs. PyTorch: A Comparative Analysis

Among the deep learning frameworks, TensorFlow and PyTorch are the most popular. Their design philosophies and cultures are distinct, so it depends on the situation. Table 5.1 provides a comparative summary of the frameworks.

For this project, TensorFlow was chosen over PyTorch due to its production-ready features and superior performance in handling large-scale embedding-based workflows. Key considerations include:

- **Scalability:** TensorFlow seamlessly scales across multiple CPUs, GPUs, and TPUs, meeting high-throughput demands.
- **Embedding Tools:** TensorFlow provides specialized libraries for creating and managing embeddings, which are essential for image similarity calculations.
- **Community and Documentation:** Its extensive documentation and large developer community expedite troubleshooting and feature implementation.
- **Deployment Readiness:** The static computation graph ensures efficient resource utilization and optimized execution for real-time applications.

Table 5.1: TensorFlow vs. PyTorch Comparison

Feature	TensorFlow [134]	PyTorch [135]
Developer	Google Brain	Facebook AI Research
Computation Graph	Static (optimized for performance and scalability)	Dynamic (optimized for flexibility and ease of experimentation)
Ease of Debugging	More challenging due to static graph compilation	Easier due to immediate feedback
Performance	Optimized for large-scale deployments; excels in distributed computing	Suitable for research and smaller-scale experiments
Scalability	Seamless scaling across CPUs, GPUs, and TPUs	Scalable, but less streamlined in distributed environments
Community Support	Large developer community, extensive documentation	Strong support among researchers, growing in production
Embedding Support	Dedicated libraries and tools for managing embeddings	Requires custom implementation or third-party libraries
Integration	Tight integration with Keras, TensorBoard, and TensorFlow Lite	Supports TorchServe and other libraries

5.1.2 TensorFlow: A Technical Overview

TensorFlow is an open-source deep learning library that was developed by the Google Brain team, renowned for its versatility in production and research settings. Its comprehensive range of tools for constructing, training, and deploying machine learning models makes it the ideal choice for this endeavor. TensorFlow's ecosystem supports all from rapid prototyping to wide-scale production deployments, and therefore, it is a robust platform to use for embedding-based image retrieval tasks.

Key Features:

- **Computational Graphs:**
  - TensorFlow employs a static computation graph that allows for efficient execution and optimization on a broad spectrum of hardware platforms. The static graph approach delivers graph-level optimizations for improved runtime performance and reduced memory footprint.
  - Each node in the graph represents a mathematical operation, while edges represent tensors (multidimensional arrays) that flow between these operations, ensuring a structured and predictable workflow.

- **Scalability:**
  - TensorFlow makes it simple to scale from single-device training to distributed training on large clusters. It is also compatible with specialized hardware accelerators such as GPUs and TPUs, which are essential in handling large volumes of image data and complex deep learning models.
  - The framework's robust distributed computing capabilities allow it to efficiently process large datasets, a critical requirement for embedding-based image retrieval systems.
- **Integration with Keras:**
  - TensorFlow has strong integration with Keras, a high-level API that provides an easier method of designing, training, and testing neural network architectures. It provides a familiar interface that accelerates model creation and experimentation.
  - Keras' modular design, combined with TensorFlow's powerful backend, enables rapid prototyping while still offering the flexibility to deploy models in production environments with minimal modifications.
- **Embedding Support:**
  - TensorFlow includes dedicated libraries and tools for creating and managing embedding layers, which are essential for converting images into compact feature vectors used in similarity calculations.
  - These tools facilitate efficient training and deployment of models that rely on high-dimensional embeddings, contributing to both faster inference and improved retrieval accuracy.
- **Deployment and Ecosystem:**
  - TensorFlow offers a comprehensive ecosystem for deployment, including TensorFlow Serving for scalable model deployment, TensorBoard for visualization and debugging, and TensorFlow Lite for deploying models on mobile and embedded devices.
  - Its wide-ranging API support and active community provide extensive documentation and tools, ensuring that both researchers and practitioners can implement state-of-the-art solutions effectively.

Overall, TensorFlow is the framework of choice for this project because it handles the computational needs and scalability concerns that are associated with embedding-based image search most effectively.

While PyTorch offers a highly flexible environment that is best suited for research, TensorFlow's emphasis on deployment efficiency, robust production-level features, and rich ecosystem make it more suitable for real-world applications [134, 136].

5.2 Models Selected and Constructed

5.2.1 Lightweight Model

This lightweight CNN model design is intended to be capable of providing useful image classification with predictive accuracy balanced with computational efficiency. The design is intentionally made straightforward so that it can continue to be flexible for different datasets and platforms, and hence it is best suited to implement on mobile and web applications.

The model begins with a convolutional layer with 32 filters of kernel size 3x3 and stride 1, and the ReLU activation function to acquire low-level features such as edges, textures, and basic shapes. The reason for the 3x3 kernel size is borrowed from the effective designs of the original CNN architectures [25, 137], where smaller filters excel at capturing fine-grained details. An additional convolutional layer of 64 filters next specializes these features and extracts still higher-level detailed information. This growth in layer number is consistent with strategies adopted by deeper networks for gradually enriching feature complexity without increasing parameters unnecessarily.

To reduce computational complexity and reduce spatial dimensions, each layer of convolution is followed by a max pooling layer. Max pooling not only reduces the computational complexity but also provides a kind of translation invariance by selecting the most significant features from each region [137]. In this design, the max pooling layers help to extract the important information from the feature maps before they are passed to the fully connected layers.

After the convolutional and pooling layers, the output feature maps are flattened into a one-dimensional vector. This vector is fed through a dense layer with 64 neurons and ReLU activation that further refines and concatenates the learned features. A dropout layer with a 50% dropout rate is applied at training time to avoid overfitting and promote generalization, a technique that has been demonstrated to be very effective at reducing model variance [138]. In addition, the network uses He initialization [139] to initialize the weights to appropriate initial values for ReLU activations, which allows for faster convergence and improved training stability.

Finally, a softmax activation layer outputs a probability distribution across all classes, so this model is well suited for multi-class classification problems. The light weight structure, with reduced numbers of parameters and less complex architecture, is well suited for those situations where inference time and memory usage are significant factors.

Figure 5.1 presents a flowchart summarizing the architecture of the lightweight CNN model.

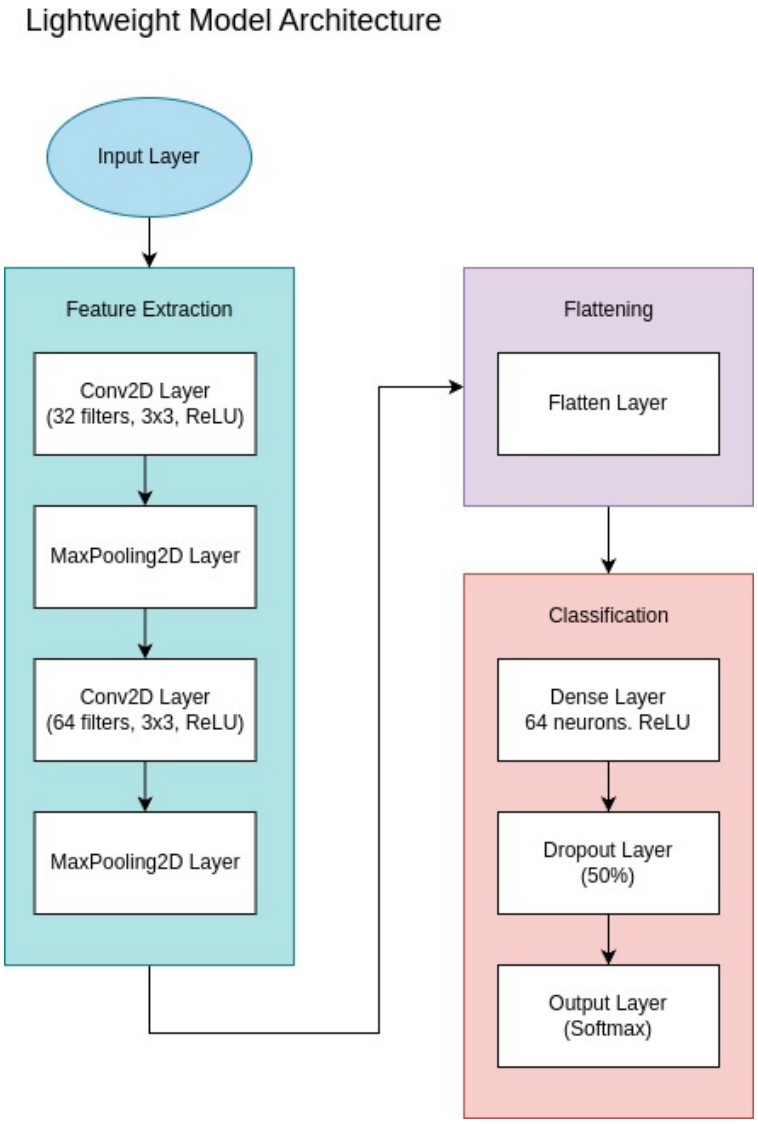

Figure 5.1: Flowchart of the Lightweight CNN Model Architecture.

5.2.2 Moderate Model

This moderate model is constructed to strike a good balance of computation and deep, complex feature extraction, and hence it is well-suited for applications that entail extensive image analysis. Following the widely used ResNet50 architecture, the model consists of multiple convolutional layers with successively larger numbers of filters to capture fine-grained patterns and maximize feature representation.

The model begins with a convolutional layer of 32 filters with a size of 3x3 and a stride of 1 followed by the ReLU activation function to learn low-level features such as texture and color information. The subsequent convolutional layers increase the number of filters step by step to 64, 128, and then 256 in order to capture increasingly finer and more complex features [28, 25]. This gradual increase allows the network to learn a hierarchical abstraction of the input information but with reduced numbers of parameters.

After each convolutional layer, max pooling with window size 2x2 is applied. Max pooling not only decreases the dimension of the feature maps to save computational expenses but also helps preserve the most outstanding features for the next processing [137]. Utilization of batch normalization (which is not actually mentioned here, but can prove useful) may also stabilize and accelerate the process of training.

Following the convolutional blocks, the feature maps are flattened into a one-dimensional vector. This vector is input through a dense layer with 512 neurons and ReLU activation, providing enough capacity for higher-level feature processing. A dropout layer with a dropout rate of 0.5 is applied after this dense layer to ensure overfitting is prevented by randomly dropping out neurons during training [138]. The network then goes on to further refine the features in a second dense layer of 256 neurons, with another dropout layer, before the final classification layer.

The output layer employs a softmax activation function to produce a probability distribution over all predefined classes, making the model suitable for multi-class classification tasks.

In summary, this moderate architecture is a nice trade-off between computational efficiency and analytical depth. Its design, several convolutional layers, dense layers, and dropout regularization, grants robustness and flexibility, allowing the model to generalize well to a range of image classification tasks while maintaining computational requirements in line.

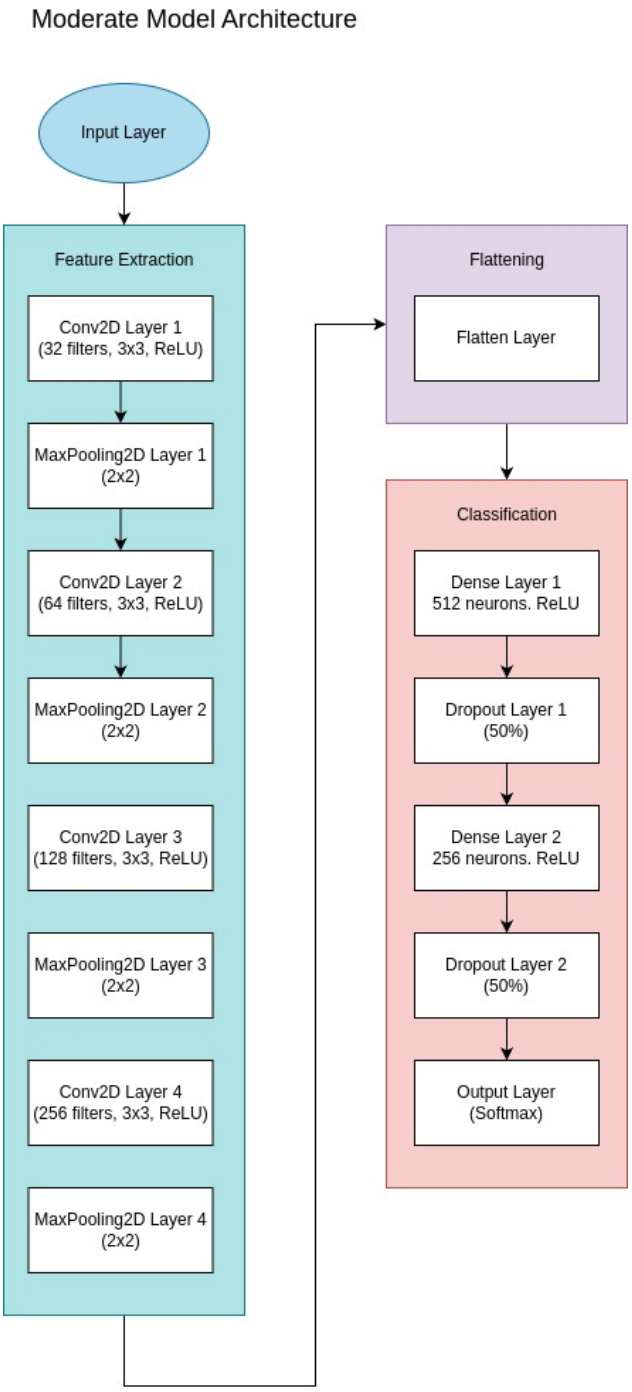

Figure 5.2: Flowchart of the Moderate CNN Model Architecture.

5.2.3 Heavy Model

This heavy model is constructed for highly accurate image classification tasks that require high levels of feature extraction. It is best suited to process complex and diverse datasets by leveraging deep convolutional layers, batch normalization, and strategic dropout, which collectively offer consistent performance even in the worst-case scenarios.

The architecture begins with a 32 filter of size 3x3 convolutional layer with ReLU activation to extract low-level features such as edges and textures. Batch normalization is employed to normalize the activations and to stabilize the learning process [140]. A second 32 filter convolutional layer with batch normalization refines the initial features further. This is followed by a max pooling layer (2x2) and a dropout layer with rate 0.25 to prevent overfitting while preserving low-level features of highest significance.

In the next phase, the filters are set to 64 so that the model can recognize more complex and subtle patterns. Two additional convolutional layers with batch normalization are employed to further process these features. Again, a max pooling layer and a dropout of 0.25 are employed to reduce the spatial dimensions and maintain computational overhead in check. This stage allows the network to build a more complex representation of the input data without being hampered by overwhelming parameter growth.

The third stage introduces even more intricacy with 128 filters to enable even more complex patterns and finer details to be separated. As in each of the preceding stages, every convolutional layer at this stage is accompanied by batch normalization, max pooling, and a dropout of 0.25 to enable stability and robust regularization across the network.

After the convolutional blocks, the model transitions to fully connected layers via flattening of the multi-dimensional feature maps into a one-dimensional vector. The one-dimensional vector is then fed into a dense layer with 512 neurons and ReLU activation, which provides high capacity for further feature refinement. To allow generalization, batch normalization is used again, followed by the use of a dropout rate of 0.5—a rate chosen in an effort to heavily mitigate overfitting during this high-capacity stage [138]. The model concludes with an output dense layer using a softmax activation function, mapping the final logits to a probability distribution over the target classes, making the model suitable for multi-class classification problems.

Key aspects of the heavy model include:

- **Complexity:** The model progressively increases the number of filters from 32 to 128, facilitating hierarchical feature extraction that captures both low-level and high-level representations.
- **Batch Normalization:** Consistently applied after each convolutional layer to stabilize and

accelerate training by reducing internal covariate shift [140].

- **Dropout:** Strategically employed with rates of 0.25 in the convolutional layers and 0.5 in the dense layers to mitigate overfitting while maintaining essential feature learning [138].
- **Fully Connected Layers:** The dense layer with 512 neurons refines the extracted features before the final classification, ensuring that the network is capable of high-level abstraction.

In summary, this heavy model delivers deep insight and detailed sophistication, making it highly effective for applications such as medical image analysis, industrial defect detection, and high-level object recognition. Its design balances computational complexity with precision, ensuring both training stability and generalizability in challenging classification scenarios.

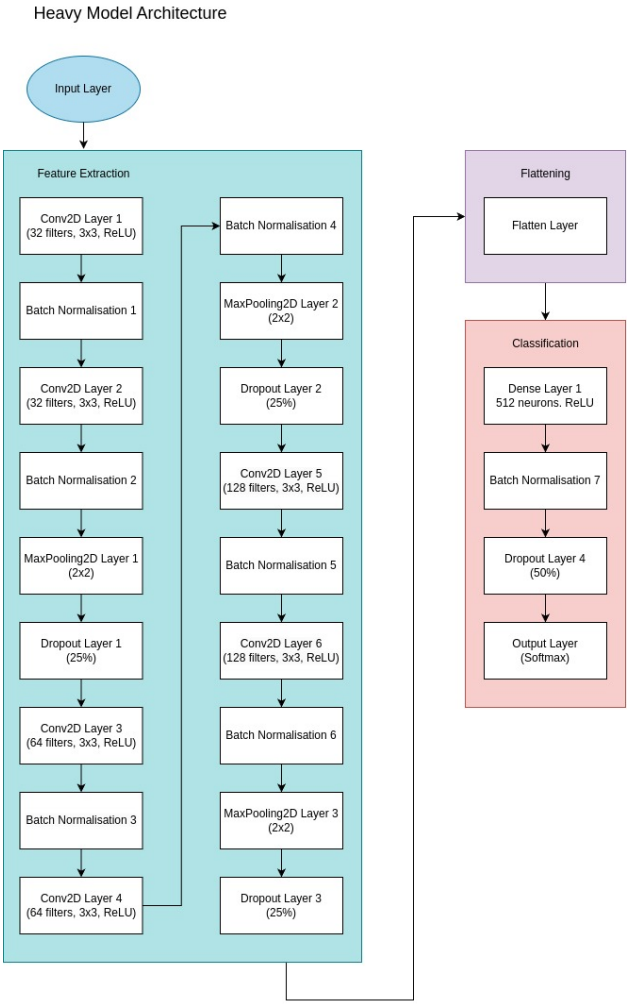

Figure 5.3: Flowchart of the Heavy CNN Model Architecture.

5.2.4 ResNet50 Model

ResNet50 is a deep convolutional neural network that has been hailed in executing complex image processing operations efficiently and resiliently. The architecture of the network utilizes residual connections, a paramount innovation that circumvents the vanishing gradient problem inherent in very deep networks, thereby enabling the network to gain depth without compromising its performance.

The network begins with the initial convolutional layer with a 7x7 kernel of 64 channels and a stride of 2. The initial set of feature maps is generated by this layer from raw pixel data. A max pooling layer with a stride of 2 and a kernel of 3x3 follows this. The spatial dimension of the feature maps is reduced through this, and therefore the amount of computation decreases, preparing the data for deeper layers.

ResNet50 is built from residual blocks. A block contains a number of convolutional layers with batch normalization and ReLU activation. Shortcut connections are included in these blocks—identity mappings when the sizes are equal, and convolutional (1x1) mappings when they are not—to enable smooth backpropagation of gradients. The basic idea behind a residual block can be represented by the equation:

$$\mathbf{y} = \mathcal{F}(\mathbf{x}, \{W_i\}) + \mathbf{x},$$

where $\mathbf{x}$ is the input to the block, $\mathcal{F}(\mathbf{x}, \{W_i\})$ represents the residual function (i.e., the stacked convolutional layers), and $\mathbf{y}$ is the output of the block [28]. As the network deepens, the number of filters increases progressively (64, 128, 256, and 512), capturing increasingly complex features.

After the residual blocks, the network employs a global average pooling layer that squeezes each of the feature maps to one single value, practically aggregating the most significant features learned. Pooling output goes to a fully connected layer to output logits for all classes. A softmax activation function then finally converts these logits to probabilities and ResNet50 is hence properly suitable for multi-class classification.

Key Features

- **Initial Layers:** A 7x7 convolutional layer and 3x3 max pooling extract initial features and reduce spatial dimensions.
- **Residual Blocks:** Stacked convolutional layers with shortcut connections (both identity and convolutional) mitigate vanishing gradients, enabling effective training of very deep networks.
- **Global Average Pooling:** Aggregates feature maps into single values that emphasize critical

features while reducing the total number of parameters.

- **Fully Connected Layer & Softmax:** Converts the pooled features into class logits and outputs a probability distribution over the target classes.

Figure 5.4 illustrates the ResNet50 layers architecture, adapted from [141, 142]. Additional details on residual learning can be found in the original ResNet paper [28], which provides a comprehensive analysis of the advantages offered by these shortcut connections.

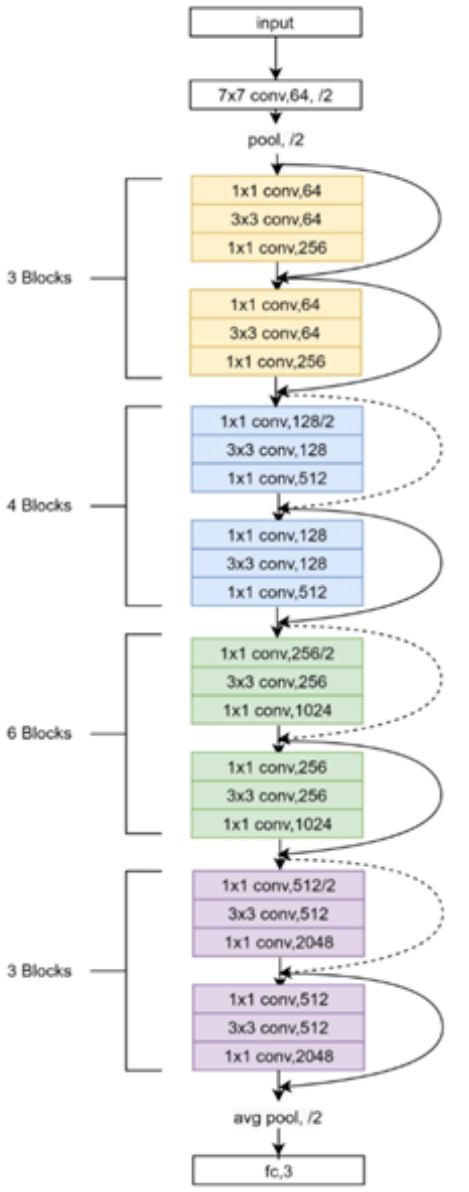

Figure 5.4: Illustration of ResNet50 layers architecture. Adapted from [141, 142].

5.2.5 MobileNetV2 Model

MobileNetV2 is a convolutional neural network architecture specifically designed for mobile and embedded vision applications. Building on the success of MobileNetV1 [143], MobileNetV2 introduces key innovations, such as inverted residual blocks with linear bottlenecks and depthwise separable convolutions, that strike a balance between computational efficiency and accuracy.

The network starts with a convolutional layer that has a 3x3 kernel with 32 output channels and a stride of 2. The layer is succeeded by batch normalization and a ReLU6 activation, which clamps the activation to six to facilitate quantization and performance on mobile devices. The key innovation of MobileNetV2 is in its inverted residual blocks. These blocks first project the low-dimensional input into a higher-dimensional space, apply transformation to the features, and then project them onto a lower-dimensional representation. When the input and output dimensions are equal, a residual (or skip) connection that directly adds the input to the output is employed to avoid losing essential information. This residual mapping can be expressed mathematically as:

$$\mathbf{y} = \mathcal{F}(\mathbf{x}) + \mathbf{x},$$

where $\mathbf{x}$ is the input to the block and $\mathcal{F}(\mathbf{x})$ represents the transformation performed by the block (i.e., the series of convolutions, non-linearities, and linear bottlenecks) [28].

One of the key characteristics of MobileNetV2 is using depthwise separable convolutions, which separate a regular convolution into two: a depthwise convolution and a pointwise convolution. For a regular convolution with kernel size $k \times k$, input channels M, and output channels N, the computational complexity is on the order of:

$$k \times k \times M \times N \times D_F \times D_F,$$

where $D_F \times D_F$ is the spatial dimension of the output feature map. In contrast, depthwise separable convolution first applies a depthwise convolution with cost:

$$k \times k \times M \times D_F \times D_F,$$

followed by a 1×1 pointwise convolution with cost:

$$M \times N \times D_F \times D_F.$$

The total cost is significantly reduced, making the architecture more efficient.

The network is organized into several stages:

- **Stage 1:** Uses an expansion factor of 1 with 16 output channels, repeated once with a stride of 1.
- **Stage 2 onward:** Employ an expansion factor of 6, with progressively increasing output channels (e.g., 24, 32, 64, 96, 160, 320) across multiple repetitions and strides to reduce spatial dimensions and deepen the network.

In the later layers, the number of channels is increased to 1280 (or 1024 in some variants) by another convolutional layer, batch normalization, and ReLU6 activation. Every feature map is then condensed to a single value by a global average pooling layer. The resulting feature vector is fed through a fully connected layer that produces logits for classification, which are finally converted to probabilities by a softmax activation function.

Key Features:

- **Depthwise Separable Convolutions:** Significantly reduce the number of parameters and computational cost by decoupling spatial and channel-wise operations.
- **Inverted Residual Blocks:** Use skip connections that preserve essential information while maintaining computational efficiency, as captured by the residual mapping equation above.
- **Linear Bottlenecks:** Prevent information loss in low-dimensional representations by omitting non-linear activations in the bottleneck layer.
- **ReLU6 Activation:** Optimized for quantization, enhancing performance on resource-constrained devices.

MobileNetV2 represents a significant advancement in neural network design for resource-constrained environments, balancing efficiency and accuracy. Its architecture makes it an excellent choice for real-world deep learning applications.

Figure 5.5 illustrates the overall architecture of the MobileNetV2 network.

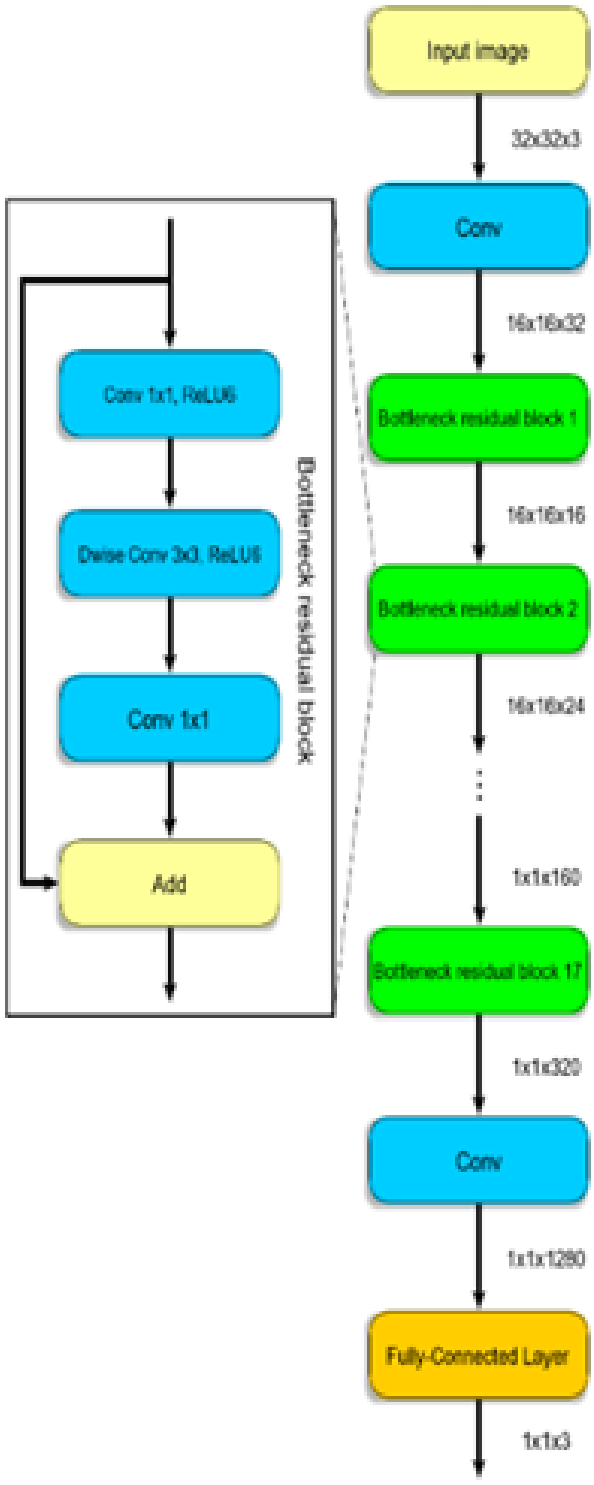

Figure 5.5: The architecture of the MobileNetV2 network. Adapted from [144].

5.2.6 EfficientNetB0 Model

EfficientNetB0 is an advanced CNN architecture that achieves state-of-the-art performance with exceptional computational efficiency. Developed by Google AI, it introduced a novel compound scaling methodology that uniformly scales network depth, width, and resolution using fixed coefficients. This balanced scaling enables EfficientNetB0 to outperform many existing models while requiring significantly fewer parameters compared with older designs [33].

EfficientNetB0 builds upon concepts from MobileNetV2 by incorporating Mobile Inverted Bottleneck Convolution (MBConv) blocks enhanced with Squeeze-and-Excitation (SE) modules. Each MBConv block consists of:

- **Expansion Layer:** Increases the number of channels to capture detailed features.
- **Depthwise Convolution:** Efficiently captures spatial relationships.
- **Projection Layer:** Reduces channels back to a lower dimensionality for computational efficiency.
- **Squeeze-and-Excitation Modules:** Recalibrate channel-wise feature responses to emphasize the most informative features.

The model starts with the initial convolutional layer having a 3x3 kernel, 32 output channels, and a stride of 2. This is then followed by batch normalization and Swish activation function, providing smoother gradients and higher expressiveness than ReLU. The network goes through multiple stages of MBConv blocks:

- **Stage 1:** An MBConv1 block with a 3x3 kernel, 16 output channels, and stride 1.
- **Stages 2 to 8:** MBConv6 blocks with varying kernel sizes (3x3 or 5x5) and progressively increasing output channels (ranging from 24 to 320), with appropriate strides to downsample spatial dimensions.

After the MBConv stages, there is a convolutional layer whose channels are increased to 1280 (1024 in some versions), followed by batch normalization and Swish activation. Then, global average pooling reduces the value of every feature map down to one number. A fully connected layer lastly produces logits to be classificated with, which is then converted to probability using softmax activation.

A key innovation in EfficientNet is its **compound scaling** method. Instead of scaling network depth, width, and resolution arbitrarily, EfficientNet scales these dimensions uniformly using a compound coefficient ϕ. This is expressed as:

$$d = \alpha^{\phi}, \quad w = \beta^{\phi}, \quad r = \gamma^{\phi},$$

subject to the constraint $\alpha \cdot \beta^2 \cdot \gamma^2 \approx 2$, where d, w, and r denote depth, width, and resolution, respectively, and α, β, γ are constant scaling factors determined via grid search [33]. This systematic scaling allows EfficientNetB0 to achieve a better trade-off between accuracy and computational cost.

Key Innovations

- **Compound Scaling:** Uniformly scales depth, width, and resolution using fixed coefficients for balanced resource usage and optimal performance.
- **MBConv Blocks:** Leverage depthwise separable convolutions and linear bottlenecks for computational efficiency.
- **Squeeze-and-Excitation Modules:** Enhance representational capacity by recalibrating channel-wise feature responses.
- **Swish Activation:** Provides smoother gradient flow and superior performance compared to ReLU.

EfficientNetB0 represents a breakthrough in neural network design by offering an innovative and efficient approach to scaling, making it ideal for real-world applications that require a balance between performance and resource constraints.

Figure 5.6 illustrates the detailed architecture of EfficientNetB0.

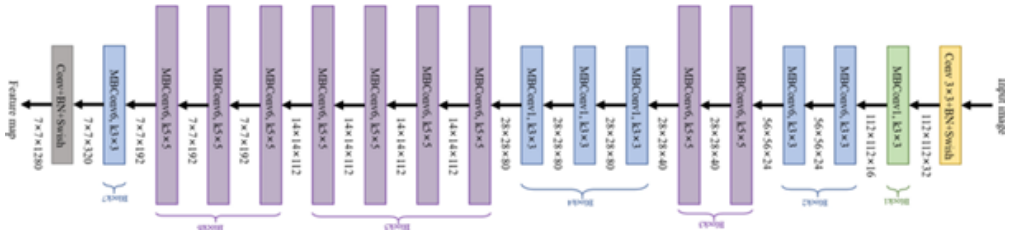

Figure 5.6: EfficientNetB0 Model Selected Architecture. Adapted from [145].

5.2.7 Other Models Considered and Limitations

In the process of developing the embedding-based image retrieval system, several other deep learning architectures were explored before finalizing on the chosen models of ResNet50, MobileNetV2, EfficientNetB0, and the custom models. The following discussion takes into account the alternatives, their positives, and the limitations that led to the final model selection decisions.

Inception Networks Inception-based architectures (e.g., GoogLeNet, Inception-v3) employ multi-scale convolutions within the same layer, allowing the network to capture spatial hierarchies at different resolutions simultaneously [146]. This approach has been shown to enhance feature diversity and improve classification performance.

Advantages:

- Multi-scale feature extraction enhances robustness to variations in object size and orientation.
- Improved accuracy on large-scale datasets such as ImageNet.

Limitations:

- Computational Overhead: The multi-branch architecture increases the number of operations per layer, requiring significantly higher GPU memory and inference time.
- Complexity in Deployment: Inception-based models demand optimized execution frameworks to maintain efficiency, making integration into a lightweight image retrieval pipeline challenging [91].

Due to these constraints, Inception architectures were deemed less suitable for real-time retrieval applications, where inference speed and memory efficiency are critical.

VGG Networks The VGG family of networks (VGG16, VGG19) is characterized by a simple and uniform deep architecture, using small (3×3) convolutional filters stacked in depth [147].

Advantages:

- Strong Baseline Performance: The deep and uniform architecture is well-studied and serves as a strong feature extractor.
- Pretrained Availability: VGG models have widely available pretrained weights, making transfer learning straightforward.

Limitations:

- High Parameter: Count VGG architectures have 138M+ parameters, making them computationally expensive [148].
- Large Memory Footprint: The high number of parameters increases VRAM requirements, making deployment on mobile and edge devices impractical.
- Slow Inference: The sequential layer structure results in higher latency compared to more optimized architectures like ResNet [149].

Despite their strong feature extraction capabilities, VGG networks were excluded due to high memory usage and slow inference speed.

DenseNet DenseNet introduces dense connectivity, where each layer is connected to every other layer in a feed-forward manner [150]. This design maximizes feature reuse while reducing redundancy.

Advantages:

- Parameter Efficiency: DenseNet achieves comparable accuracy with fewer parameters than VGG.
- Improved Gradient Flow: Direct connections between layers mitigate vanishing gradient issues, facilitating deeper networks.

Limitations:

- Increased Memory Usage: While parameter-efficient, the dense connections increase activation memory requirements, posing challenges for training and deployment on GPUs with limited VRAM [93].
- Computational Complexity: Feature concatenation at every layer introduces higher computational overhead, affecting inference speed.

Although DenseNet provides high accuracy, its increased memory usage and computational complexity made it less practical for our real-time retrieval pipeline.

Custom Lightweight Architectures Various custom lightweight CNN architectures were explored to optimize speed and efficiency. These models were designed to minimize FLOPs (floating point operations) and parameter count, reducing computational cost.

Observations:

- Lightweight models performed well in general classification tasks but failed to capture deep feature representations necessary for fine-grained image retrieval [91].
- Lack of pretrained weights made training more data-intensive compared to established models like ResNet and EfficientNet.

While custom models were promising for efficiency, they lacked the feature depth necessary for medical imaging and domain-specific retrieval tasks.

The final model selection was based on a trade-off between accuracy, computational efficiency, and deployment feasibility:

- **ResNet50** offers robust feature extraction with its residual connections, ensuring high accuracy even in complex image retrieval tasks.
- **MobileNetV2** and **EfficientNetB0** provide an optimal balance between efficiency and accuracy, making them ideal for real-time applications on resource-constrained devices.
- **Custom models** were fine-tuned to meet specific domain requirements, ensuring flexibility in handling specialized datasets.

While architectures such as Inception, VGG, and DenseNet are also satisfactory, the demands of real-time performance, computational efficiency, and domain-specific accuracy of the application made it clear to select ResNet50, MobileNetV2, and EfficientNetB0. Ensemble methods or hybrid models can be studied in the future to further enhance retrieval performance.

5.3 Image Retrieval Methods

Image retrieval refers to the search and retrieval of images from a dataset ordered based on similarity to a query image. Using embeddings and similarity measures, the retrieval system returns the most similar images. Two primary methods of implementing image retrieval in this project are FAISS and K-Nearest Neighbors (KNN).

5.3.1 K-Nearest Neighbors (KNN)

KNN is a straightforward yet effective algorithm for image retrieval based on embeddings. It involves finding the K most similar embeddings to a query embedding by calculating similarity or distance metrics such as Euclidean distance or cosine similarity [151, 152].

Key Features:

- **Simplicity:** KNN is easy to implement and does not require any prior model training.
- **Distance-Based Search:** It retrieves neighbors by comparing distances between embeddings, making it flexible with various similarity metrics.
- **Scalability Challenges:** For very large datasets, KNN becomes computationally expensive because it requires pairwise comparisons between the query and all embeddings [152, 153].

5.3.2 FAISS (Facebook AI Similarity Search)

FAISS is an efficient library developed for large-scale similarity search and clustering of dense embeddings. It is particularly well-suited for high-dimensional data, such as image embeddings [154, 155].

Key Features:

- **Optimized for Scalability:** FAISS employs advanced indexing structures such as the Inverted File Index (IVF) and Product Quantization (PQ) to efficiently handle billions of embeddings.
- **GPU Acceleration:** FAISS supports GPU-based computations, which significantly speed up search operations.
- **Customizable Indexing:** It provides multiple indexing techniques, allowing users to balance speed and retrieval accuracy based on their specific requirements.

5.3.3 Comparison of KNN vs. FAISS

Table 5.2 summarizes a comparison between KNN and FAISS in terms of scalability, speed, ease of use, and flexibility.

Table 5.2: Comparison of KNN vs. FAISS

Aspect	KNN	FAISS
Scalability	Limited by dataset size; slower for large datasets.	Optimized for large-scale datasets; supports billions of vectors.
Speed	Relatively slow for large datasets.	Fast due to advanced indexing and GPU support.
Ease of Use	Simple to implement.	Requires setup and configuration but offers extensive features.
Flexibility	Works with various distance metrics.	Customizable indexing with advanced trade-offs.

Both KNN and FAISS are solid image retrieval algorithms. KNN is straightforward and handy for small datasets, but FAISS takes the lead in large scale due to its advanced indexing data structures and GPU acceleration [154, 155].

5.3.4 Other Retrieval Methods Considered and Limitations

Apart from KNN and FAISS, other retrieval techniques were also explored to verify their peformance and usage in this application setting. However, owing to the tests for performance and hardware constraints, these were not selected for the final retrieval system.

Approximate Nearest Neighbors (ANN) Algorithms: A number of ANN algorithms (i.e., Annoy [156] and FLANN [157]) were considered due to their ability to provide efficient retrieval times at the expense of large sets. While such methods are competitive in performance, they sacrifice precision for speed. In the experiments, the lowest loss of retrieval precision was not acceptable in light of our application's precision requirements.

Hashing-based Methods: Such methods as Locality Sensitive Hashing (LSH) [158] were experimented with. LSH can drastically accelerate search time by mapping high-dimensional data to short binary codes. But in the process of transformation, loss of information occurs, resulting in low retrieval quality, particularly when fine-grained discrimination is called for.

Graph-based Methods: Graph-based retrieval methods construct a similarity graph among data points and perform search by walking on the graph [159]. Although such methods can obtain high retrieval precision, they are typically hard to implement and require much memory and thus are less suitable for real-time applications.

The final selection of KNN and FAISS was driven by an equally weighted balance between accuracy, computational expense, and ease of integration. KNN, despite being computation-constrained for very large datasets, possesses the simplicity and flexibility that are appropriate for small to medium-scale applications. FAISS, on the other hand, boasts greatly enhanced scalability and GPU capabilities, and is thus appropriate for large-scale retrieval tasks. The other methods, promising as they were in certain conditions, were unable to meet the overall needs of high precision at affordable resource use and deployment convenience.

5.4 Image Retrieval Pipelines

5.4.1 Domain-Specific Retrieval Pipeline (DSRP)

Pipeline Construction and Overview

The Domain-Specific Retrieval Pipeline is an end-to-end image retrieval pipeline that facilitates effective and efficient search using K-Nearest Neighbors or Facebook AI Similarity Search (FAISS). The pipeline offers the possibility of using custom-trained models (specific to the task at hand) as well as widely used pre-trained models such as ResNet50, EfficientNetB0, and MobileNetV2. The models can be fine-tuned on the destination dataset for learning domain-specific features. This two-forked approach offers great flexibility and improved retrieval accuracy in diverse application fields. The entire pipeline is conceived to be modular, clear, and highly scalable, making each component comprehensible, extensible, and adaptable to various types of data and hardware setups [160].

Pipeline Architecture

DSRP is set up in a modular and transparent manner. Each of the configuration parameters such as data path, model selection, fine-tuning flags, image size, neighbors count, retrieval method (KNN or FAISS), and device mode (CPU or GPU) are managed by a YAML config file. It allows for simple modification without changing the underlying codebase.

The pipeline dynamically modifies the computation context by perceiving and utilizing GPU resources when they are available, otherwise falling back to CPU, thus enabling support for heterogeneous hardware configurations. Regarding model selection, DSRP supports off-the-shelf structures (e.g. ResNet50, EfficientNetB0, MobileNetV2) as well as user-trained models. Models are fine-tuned on domain-adapted data for enhanced ability in feature extraction, which is crucial to acquire high retrieval precision [161].

In the pipeline, training images are resized, preprocessed, and mapped to feature vectors with robust processing for unreadable images or NaN values. Test images are treated in a typical manner to guarantee completeness of classes. Either FAISS or KNN is utilized for building a retrieval index by normalizing feature vectors for precision and efficiency. In retrieval, query images are matched against the index and performance is evaluated using measures such as majority voting and Top-N accuracy. Finally, the pipeline logs performance metrics such as inference time, CPU, and memory usage and offers valuable insight into the computational efficiency and real-world utility of the

system [162].

Flowchart

Figure 5.7 shows the flowchart of the Domain-Specific Retrieval Pipeline (DSRP). The diagram outlines key components of the pipeline—from data ingestion and preprocessing to feature extraction, index building, and retrieval evaluation, illustrating its modular and transparent design.

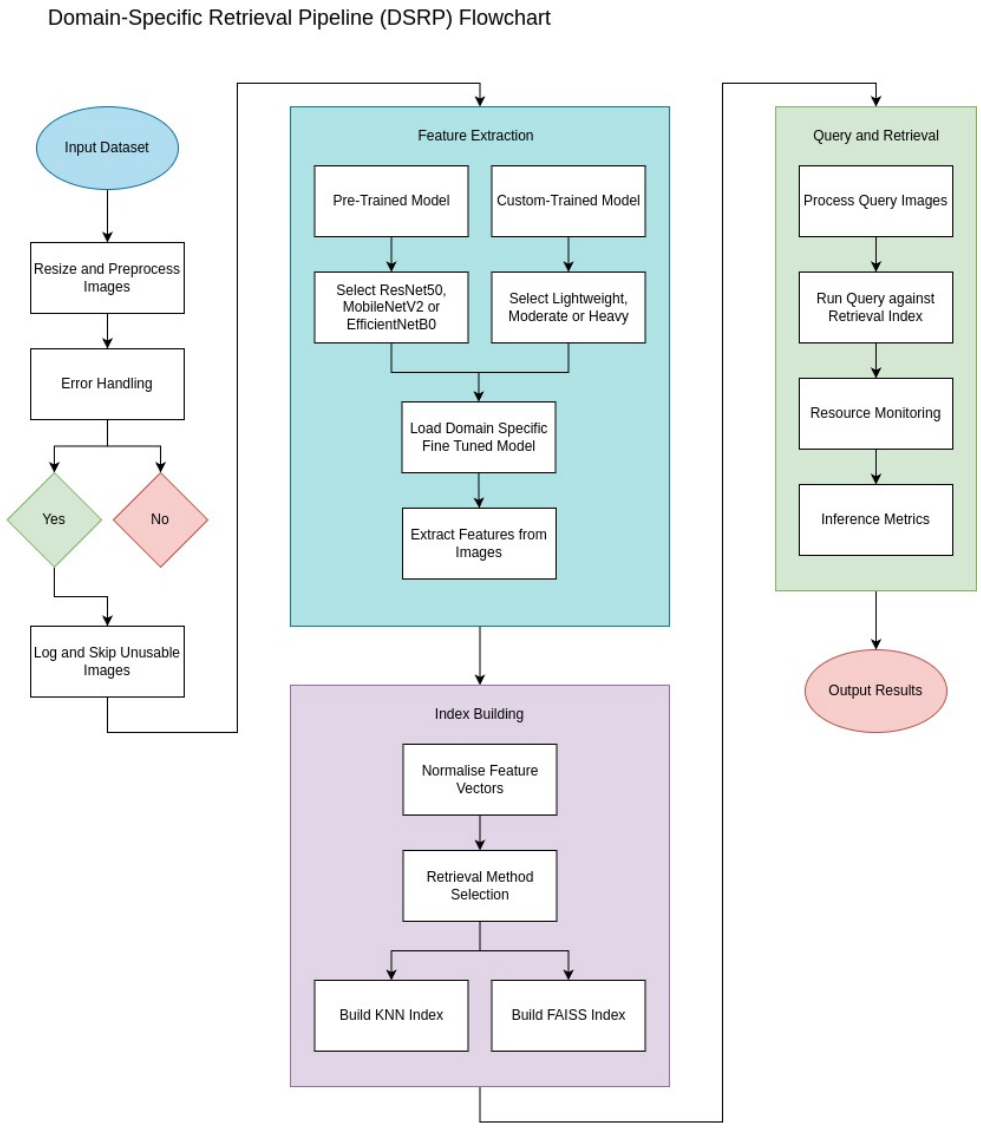

Figure 5.7: Domain-Specific Retrieval Pipeline (DSRP) Flowchart.

5.4.2 Pre-trained Feature Extraction Pipeline (PMRP)

Pipeline Construction and Overview

The Pre-trained Model Retrieval Pipeline (PMRP) is a lightweight and efficient image retrieval system based solely on pre-trained models without fine-tuning. In contrast with the Domain-Specific Retrieval Pipeline (DSRP), PMRP leverages general-purpose embeddings of widely used architectures such as ResNet50, EfficientNetB0, and MobileNetV2. PMRP is designed for rapid deployment and scalability and can be used for real-time applications or when domain-specific fine-tuning is infeasible. The PMRP focuses on simplicity, resource usage, and configuration ease while supporting heterogeneous hardware environments [160].

Pipeline Architecture

The PMRP is constructed with a modular design that promotes consistency and transparency. Key components include:

- **Configuration Management:** A YAML configuration file controls essential parameters, such as dataset paths, model selection, image size, retrieval method (KNN or FAISS), and device type (CPU or GPU), which makes the pipeline highly adaptable without modifying the underlying code.
- **Hardware Resource Setup:** The pipeline dynamically configures computational resources, automatically detecting and utilizing GPUs if available, and defaulting to CPU otherwise. This ensures broad hardware compatibility.
- **Pre-trained Model Usage:** Unlike DSRP, PMRP relies exclusively on pre-trained models from TensorFlow Keras applications (e.g., ResNet50, EfficientNetB0, MobileNetV2) to extract general-purpose feature embeddings from both training and test images.
- **Preprocessing:** Training images are pre-processed through resizing, normalization, and passing through the selected pre-trained model to obtain feature embeddings. Test images are processed in the same way to ensure consistency.
- **Indexing and Retrieval:** Depending on the retrieval method specified (KNN or FAISS), the pipeline builds a retrieval index using normalized feature vectors. Query test images are then matched against this index, and performance is measured using accuracy metrics (e.g., majority voting and Top-N accuracy).

- **Performance Monitoring:** The pipeline also tracks computational performance metrics such as average inference time, CPU, and memory usage to assess efficiency.

Flowchart

Figure 5.8 illustrates the overall architecture and workflow of the Pre-trained Model Retrieval Pipeline (PMRP). This flowchart outlines the modular steps—from configuration and preprocessing through embedding extraction, index building, and retrieval evaluation—highlighting the streamlined design intended for resource-efficient, large-scale or real-time applications.

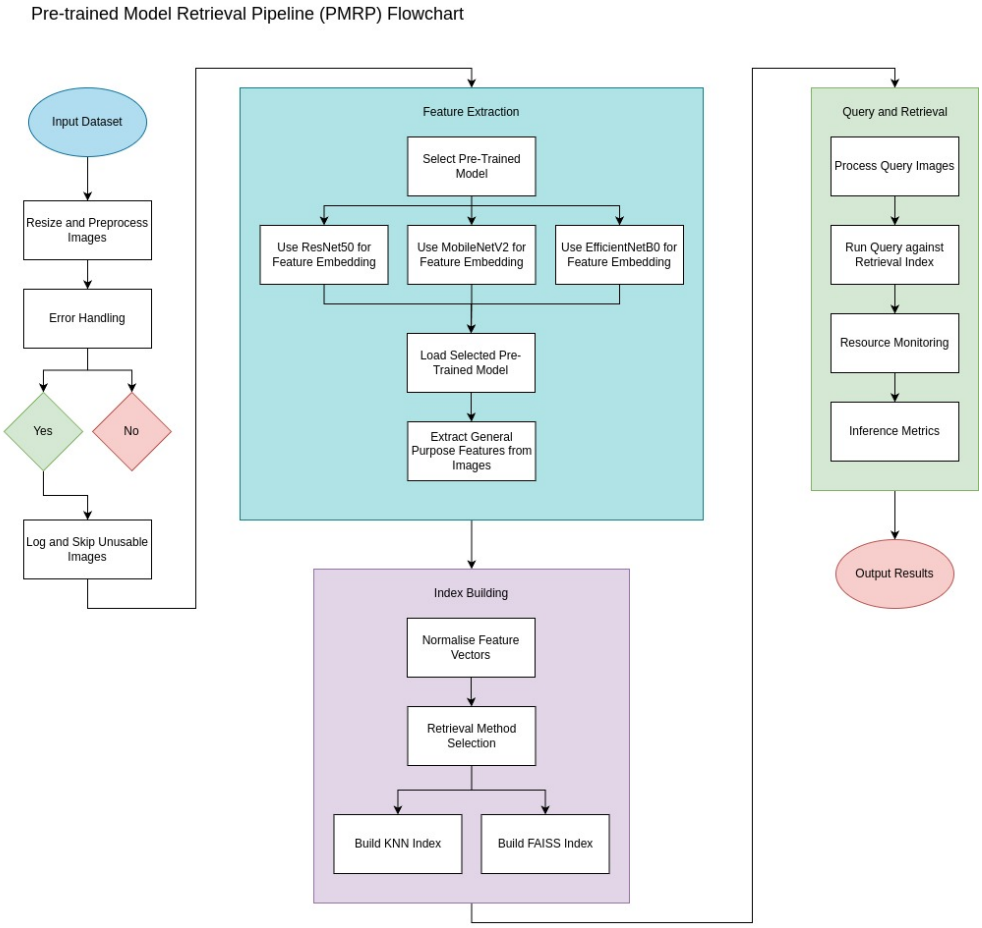

Figure 5.8: Pre-trained Model Retrieval Pipeline (PMRP) Flowchart.

Chapter 6

Experimental Setup and Results

6.1 Overview

Experimental evaluation was developed on a solid test across two image search pipelines, i.e., Pre-trained Feature Extraction Pipeline (PMRP) and Domain-Specific Retrieval Pipeline (DSRP), on four various datasets divided across healthcare, agriculture and general domains, such as the Pokemon dataset. Setup consisted of current deep learning architecture models for feature extraction, involving fine-tuning and background removal approaches for domain specificity. Furthermore, FAISS indexing was introduced to enhance computation efficiency through vast reductions in inference times with complete preservation of accuracy [154].

Each dataset, ranging from the Leaf Disease [95] and Brain Tumour MRI scans [96] to the COVID-19 Digital X-Ray Forgery [97] and Pokémon benchmark [98], was hand selected in such a way as to represent subtle challenges such as class imbalance, suble visual changes, and forgery detection [41, 42, 152]. The experiment demonstrated that DSRP provides consistently higher precision for domain-specific applications while PMRP can serve as a robust, lightweight substitute for general retrieval instances. In conclusion, the experiments validate the effectiveness of embedding-based retrieval systems, highlighting their scalability, effectiveness, and applicability to real-world applications in high-stakes environments [160, 162].

6.2 Hyperparameter Configuration

Hyperparameter selection is an essential deep learning model optimization element, which affects training stability, convergence speed, and generalizability of the overall model [163]. For stable evaluation on all datasets, a 70/15/15 train/validation/test split was used. Empirically, this ratio was chosen following best practices to leave enough data for model learning as well as for testing generalization [164].

Training Constraints and Epoch Selection

Due to hardware constraints, dataset heterogeneity, and the broad range of models tested, training was restricted to 20 epochs per model. This comparatively modest duration was used to trade computational practicability against sound benchmarking.

Reasoning for 20 Epochs Selection:

- Studies have shown that most deep learning models exhibit rapid improvements in the first 10–20 epochs, after which performance gains begin to plateau [165].
- Longer training durations improve accuracy but come at the cost of increased computation time and GPU energy consumption, which is a major concern in large-scale retrieval systems [166].

Implications of Limited Epochs:

- Faster benchmarking allowed for comparative model evaluation across different architectures.
- While additional training could yield slight accuracy gains, the objective was to identify the most efficient models rather than overfit on specific datasets.

Optimization Techniques Used

To counteract the limited training epochs, several hyperparameter tuning strategies were employed:

Learning Rate Scheduling:

- Adaptive learning rates using a cosine decay schedule helped prevent overshooting optima.
- This approach was found to be computationally more efficient than static learning rates [167].

Batch Size Selection:

- Batch sizes were set to 32 or 64, balancing GPU memory constraints with gradient stability.
- Studies have shown that smaller batch sizes (e.g., 16) result in noisier gradients, while larger batches (e.g., 128) increase memory usage with diminishing accuracy gains [168].

Regularization and Dropout:

- Dropout rates of 0.2–0.3 were used to reduce overfitting, which is particularly beneficial for small datasets.
- L2 weight decay helped improve generalization by penalizing large weights.

These hyperparameter choices were informed by empirical studies that emphasize the trade-off between model accuracy and computational efficiency in deep learning applications [169].

6.3 Resource Consumption

The experiments were conducted on a high-performance workstation equipped with an Intel® Core™ i9-10885H CPU running at 2.40GHz, 32 GB of RAM, and an NVIDIA Quadro RTX 3000 with Max-Q Design. This configuration provided a robust balance between CPU and GPU capabilities.

The resource usage estimation in image retrieval pipelines differs with changes in hardware settings, background concurrent tasks, and inherent workload fluctuations [170]. The values of resource usage presented here are averages over several runs and intended to provide general insights rather than precise measurements.

Variability in Measurements

Resource utilization is highly dependent on the following:

- Background Processes: System-level tasks and I/O operations affect available CPU/GPU power.
- GPU Memory Allocation: Memory fragmentation can lead to VRAM underutilization or overflow errors.
- Dynamic Workloads: Batch size, model complexity, and dataset size all contribute to variations in inference speed and training efficiency.

To mitigate fluctuations, all experiments were conducted under controlled conditions, with background processes minimized where possible.

Overhead from Additional Scripts

Execution of preprocessing, logging, and visualization scripts introduces additional CPU/GPU load, which must be accounted for when analyzing efficiency.

Logging Mechanisms:

- GPU/CPU usage was monitored using NVIDIA's System Management Interface (nvidia-smi).
- Real-time memory tracking ensured that VRAM bottlenecks were detected early.

Data Preprocessing Costs:

- Feature extraction and data augmentation increase CPU usage, especially for large image datasets.

Approach to Reporting Resource Use

To ensure consistent reporting, all resource metrics were averaged across multiple experimental runs, accounting for minor fluctuations caused by hardware variations and system-level processes [165].

Key Metrics Considered:

- **Inference Time:** Average time per query or batch retrieval.
- **Hardware Utilization:** Peak and average GPU/CPU usage during training and inference.
- **Memory Usage:** RAM and VRAM consumption over multiple runs.

Findings on Resource Consumption

Training Efficiency:

- Models with higher parameter counts (e.g., ResNet50, DenseNet) required more GPU resources but converged faster.
- MobileNetV2 and EfficientNetB0 demonstrated lower memory requirements, making them ideal for resource-limited devices.

Inference Performance:

- ResNet-based architectures exhibited higher GPU utilization (80–90%) but provided better retrieval accuracy.
- Lightweight models (MobileNetV2, EfficientNetB0) achieved lower inference latency, consuming 30–40% less memory than heavier models.

While these findings highlight the importance of balancing computational efficiency with accuracy, they should be interpreted as approximations due to variations in real-world deployment environments [166].

6.4 Analysis of Model Architectures

The neural network structure, how many parameters and layers, for example is central to their ability to learn sophisticated features, their ability to support wide ranges of datasets, and their ability to generalize. These characteristics are outlined below and their utility in image retrieval is addressed.

Comparison of Selected Models

Table 6.1 summarizes a comparison of the number of parameters and layers for selected models.

Table 6.1: Comparison of Number of Parameters and Layers for Selected Models

Model	Number of Parameters	Number of Layers
Lightweight	~12,000,000	8
Moderate	~19,000,000	14
Heavy	~38,000,000	23
ResNet50	~26,000,000	50
MobileNetV2	~3,400,000	53
EfficientNetB0	~5,300,000	82

Figures 6.1 and 6.2 show plots of the number of parameters and layers for each model, respectively.

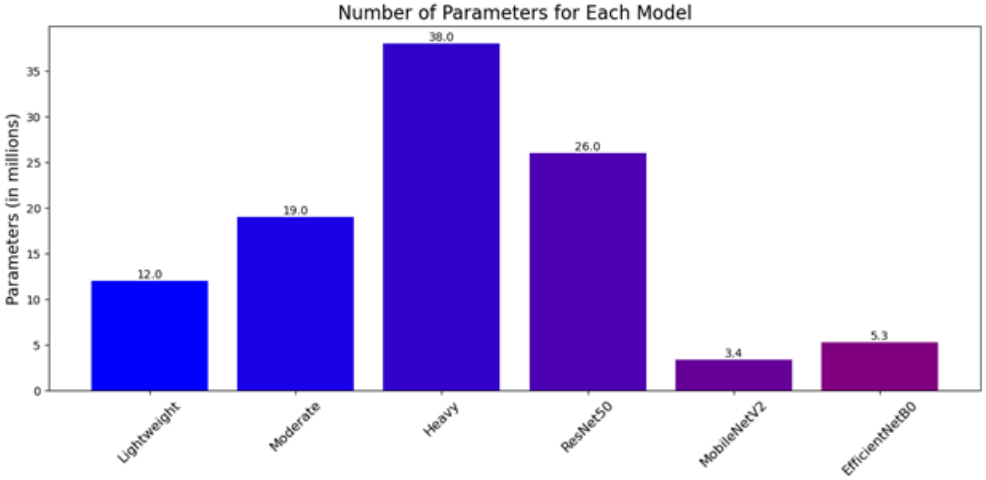

Figure 6.1: Plot of Number of Parameters for Each Model.

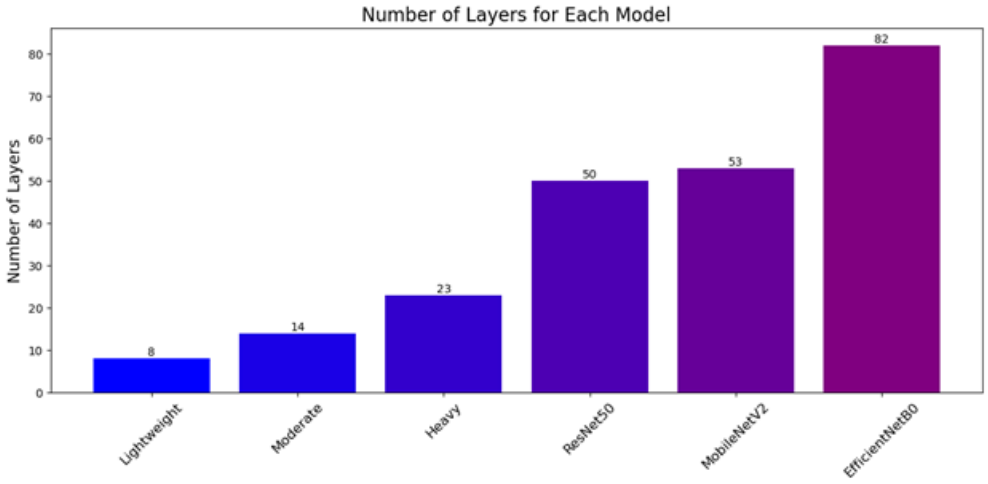

Figure 6.2: Plot of Number of Layers for Each Model.

Lightweight Model

It can seen that the lightweight model has a high parameter count because the architecture contains fully connected Dense layers after convolutional and pooling operations. The Dense layer with the highest contribution is directly connected to the Flatten layer, which is the first Dense layer. For an input of 224x224 image size, assume that the feature map output of the second MaxPooling layer is 56x56 with 64 channels, i.e.,

$$56 \times 56 \times 64 = 200\,704 \text{ features.}$$

These features are flattened and fed into a Dense layer of 64 units. The number of parameters in this Dense layer is:

$$(200\,704 + 1) \times 64 \approx 12\,845\,440 \text{ parameters.}$$

On the other hand, the convolutional layers have relatively fewer parameters to account for e.g., 896 for the first and 18,496 for the second. Without optimizations such as Global Average Pooling from [171], which reduces dimensionality before the Dense layers, the number of parameters explodes. While the architecture is straightforward, it renders the resulting model less parameter efficient than models such as MobileNet that utilize depthwise separable convolutions in an effort to reduce parameters drastically [143].

Moderate Model

The moderate model has a deeper architecture with additional convolutional layers and wider Dense layers. Its four convolutional layers have filter counts of 32, 64, 128, and 256, respectively. For

example, the parameters for these layers are computed as:

- First layer: $(3 \times 3 \times 3 + 1) \times 32 = 896$,
- Second layer: $(3 \times 3 \times 32 + 1) \times 64 = 18\,496$,
- Third layer: $(3 \times 3 \times 64 + 1) \times 128 = 73\,856$,
- Fourth layer: $(3 \times 3 \times 128 + 1) \times 256 = 295\,168$.

After these convolutional layers, assume the feature map is reduced to $14 \times 14 \times 256 = 50\,176$ features, which are then flattened. The first Dense layer, with 512 units, contributes:

$$(50\,176 + 1) \times 512 \approx 25\,690\,624 \text{ parameters.}$$

Subsequent Dense layers add more parameters e.g., the second Dense layer with 256 units adds $(512 + 1) \times 256 = 131\,328$ parameters, and the final Dense layer adds $(256 + 1) \times N$ parameters for N output classes. Although the deeper architecture adds representational capacity, it significantly increases the number of parameters and cost of computation, and the model will be prone to overfitting unless suitably regularized.

Heavy Model

The heavy model introduces additional complexity with additional convolutional layers, Batch Normalization [140], and a large Dense layer. Each convolutional block consists of two Conv2D layers with Batch Normalization and MaxPooling. Batch Normalization layers add a few trainable parameters per channel for scale and shift but are computationally inexpensive.

In the first convolutional block, two 3×3 Conv2D layers with 32 filters contribute:

- First layer: $(3 \times 3 \times 3 + 1) \times 32 = 896$,
- Second layer: $(3 \times 3 \times 32 + 1) \times 32 = 9\,248$.

Subsequent blocks with 64 and 128 filters add:

- Second block: $(3 \times 3 \times 32 + 1) \times 64 = 18\,496$ and $(3 \times 3 \times 64 + 1) \times 64 = 36\,928$,
- Third block: $(3 \times 3 \times 64 + 1) \times 128 = 73\,856$ and $(3 \times 3 \times 128 + 1) \times 128 = 147\,584$.

After these layers, MaxPooling reduces the feature map to $28 \times 28 \times 128 = 100\,352$ features. These are flattened and passed to a Dense layer with 512 units, contributing:

$$(100\,352 + 1) \times 512 \approx 51\,262\,464 \text{ parameters.}$$

The final Dense layer adds $(512 + 1) \times N$ parameters for N output classes. While Batch Normalization improves training stability and Dropout reduces overfitting, the fat model becomes overly computationally expensive because of its reliance on heavy Dense layers. Models such as MobileNet and EfficientNet use depthwise separable convolutions and Global Average Pooling to achieve significantly more cost-effective parameter usage [143, 33].

ResNet50

ResNet50 is a widely adopted deep convolutional neural network that employs residual learning to ease the training of very deep networks. The use of *skip connections* allows gradients to bypass several layers, effectively mitigating the vanishing gradient problem. Key characteristics include:

- **Architecture:** Consists of 50 layers organized into residual blocks.
- **Parameters:** Approximately 26 million parameters.
- **Design:** Utilizes a bottleneck design with 1×1 convolutions to reduce dimensions, followed by 3×3 convolutions and another set of 1×1 convolutions to restore dimensions.

The residual connections enable the network to learn identity mappings easily, improving both convergence and performance on complex tasks such as image retrieval.

MobileNetV2

MobileNetV2 is optimized for mobile and embedded vision applications, where computational resources and power are limited. Its design focuses on parameter and computation efficiency by employing:

- **Architecture:** Uses inverted residual blocks and depthwise separable convolutions.
- **Parameters:** Contains roughly 3.4 million parameters spread over 53 layers.
- **Efficiency:** The use of depthwise separable convolutions drastically reduces the number of parameters and computational cost compared to traditional convolutions.

This model strikes a balance between performance and efficiency, making it a strong candidate for real-time image retrieval tasks where low latency is crucial.

EfficientNetB0

EfficientNetB0 introduces a novel compound scaling method that uniformly scales network depth, width, and resolution. This systematic approach allows EfficientNetB0 to achieve high accuracy with fewer parameters:

- **Architecture:** Built using a baseline network that is scaled up using compound scaling.
- **Parameters:** Approximately 5.3 million parameters across 82 layers.
- **Design Innovations:** Incorporates squeeze-and-excitation optimization and employs Global Average Pooling to reduce the parameter count before classification.

EfficientNetB0's compound scaling provides a strong performance-to-parameter ratio, making it effective for both image classification and retrieval tasks while maintaining computational efficiency.

This evaluation highlights the trade-off between model complexity and computational cost. Light models are easier to train and offer faster inference but may lag in their capability to capture complex patterns, whereas heavier models can capture complex features but at the expense of greater computational costs and possible overfitting.

6.5 Introduction to the Retrieval Pipeline Results

In this section, a detailed analysis of the retrieval performance for the selected datasets (i.e., Leaf Disease, Brain Tumour MRI Scans, COVID-19 Scans, and Pokemon) using both the Domain-Specific Retrieval Pipeline (DSRP) and the Pre-Trained Feature Extraction Pipeline (PMRP) are presented.

The evaluation is based on key metrics such as Top-10 accuracy and F1 scores calculated at ranks 5 and 10 (F1@5 and F1@10). The Top-10 accuracy indicates the proportion of cases where the correct label appears within the top 10 retrieved results, which is a critical measure of the retrieval system's robustness. Similarly, the F1@5 and F1@10 metrics provide a balanced view of the precision and recall performance at these specific retrieval cutoffs.

Furthermore, this section explores the impact of background presence on the retrieval performance by comparing scenarios with and without background removal. Different models, ranging from lightweight to heavy as well as deep learning architectures, such as ResNet50, MobileNetV2, and EfficientNetB0, are analyzed using two retrieval strategies, KNN and FAISS. In addition to accuracy metrics, the inference time performance and hardware usage are assessed, offering a holistic view of both the computational efficiency and scalability of the proposed methods.

The subsequent subsections detail the performance outcomes under each condition, highlighting cases where certain models benefit from background removal and the consistent efficiency gains observed with FAISS over KNN. This comprehensive analysis helps in understanding not only the effectiveness of the models but also the practical considerations for deploying such systems.

6.6 Retrieval Pipeline Results

6.6.1 Domain-Specific Retrieval Pipeline (DSRP)

Leaf Disease (6.5.1.1)

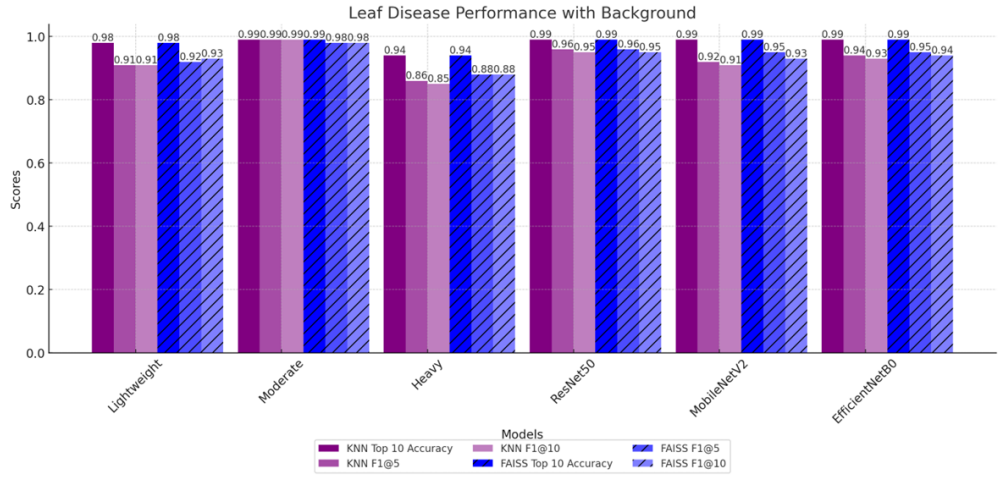

Figure 6.3: Plot showcasing Leaf Disease performance with background present using KNN and FAISS – DSRP Retrieval.

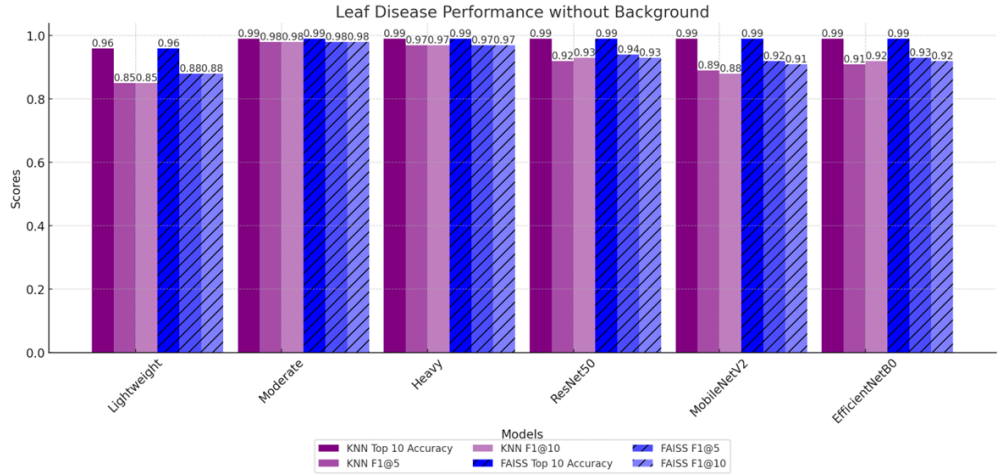

Figure 6.4: Plot showcasing Leaf Disease performance with no background using KNN and FAISS – DSRP Retrieval.

Retrieval Results The above graphs (Figures 6.3 and 6.4) shows the performance of the Leaf Disease dataset under two conditions, background present and background removed, on both KNN and FAISS. The result is that the moderate model works extremely well as shown by a Top-10

accuracy of 0.99 and good precision, recall, and F1 values (approximately 0.98–0.99) regardless of background presence or retrieval method. Contrary to this is the case with the lightweight model, where it achieves Top-10 accuracy of 0.98 with and 0.96 without background, while the heavy model shows considerable improvement without background (accuracy increases from 0.94 to 0.99, precision and recall being roughly 0.96).

Additionally, the ResNet50 model showcases robust performance with a accuracy of 0.99, recall 0.99, and precision 0.90 without background. The MobileNetV2 and EfficientNetB0 models also achieve promising results with accuracies of 0.99, although MobileNetV2 experiences minor declines in precision and recall, approximately 0.86–0.89, without background. Overall, EfficientNetB0 generalizes well across conditions, and FAISS consistently offers marginal improvements in F1 scores for most models.

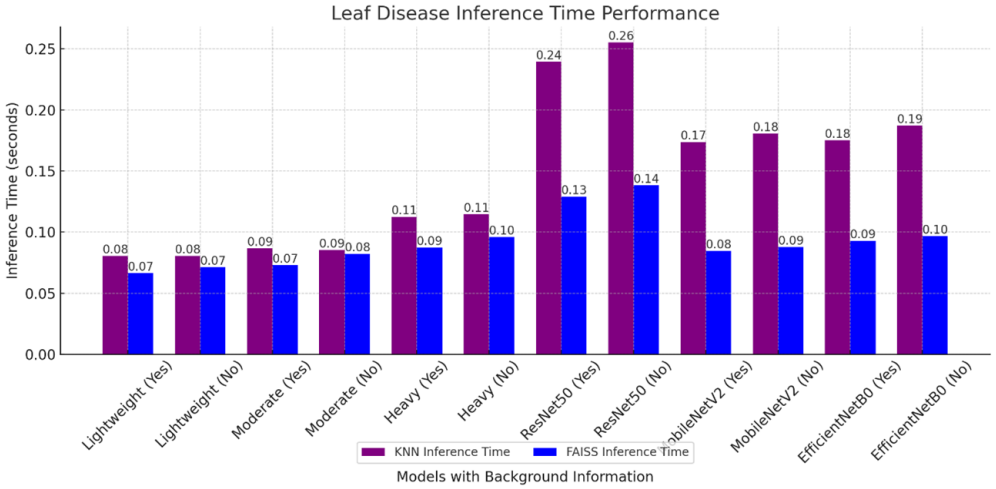

Figure 6.5: Plot showcasing the Leaf Disease inference time performance for background and no background conditions using KNN and FAISS – DSRP Retrieval.

Inference Time Performance Figure 6.5 showcases the inference time performance for the Leaf Disease dataset. For the lightweight model, KNN takes approximately 0.0806 seconds with background compared to 0.0665 seconds with FAISS, and without background, KNN takes 0.0807 seconds versus 0.0711 seconds with FAISS. Similar improvements are observed across the the different models i.e. the moderate, heavy, and deep learning models (ResNet50, MobileNetV2, EfficientNetB0), confirming FAISS consistently reduces the inference time.

Table 6.2: Hardware Usage Comparison for KNN and FAISS in DSRP Retrieval (Leaf Disease)

Model	Background	KNN		FAISS	
		CPU (%)	Memory (%)	CPU (%)	Memory (%)
Lightweight	Yes	4.8	47.6	0.1	48.1
Lightweight	No	2.7	50.1	2.5	56.8
Moderate	Yes	4.4	49.3	1.9	53.1
Moderate	No	3.6	51.3	2.8	57.9
Heavy	Yes	3.7	49.2	0.2	50.9
Heavy	No	3.7	54.2	2.4	56.7
ResNet50	Yes	2.7	50.6	0.4	52.9
ResNet50	No	4.1	53.2	2.8	56.8
MobileNetV2	Yes	4.3	50.7	0.3	50.4
MobileNetV2	No	7.9	52.0	0.8	55.8
EfficientNetB0	Yes	3.6	47.4	1.5	50.4
EfficientNetB0	No	3.5	52.0	0.3	55.1

Hardware Usage Table 6.2 summarizes the hardware usage for KNN and FAISS across different models and background conditions. The resource usage results indicate that FAISS reduces CPU usage compared to KNN across most models, particularly when the background is present. For the lightweight model, CPU usage drops from 4.8% with KNN to 0.1% with FAISS when background is present. Similar trends are observed in the moderate, heavy, and deep learning models (ResNet50, MobileNetV2, EfficientNetB0). The memory usage remains relatively stable, varying between 47.4% and 57.9% across all models and background conditions.

These results demonstrate that while the moderate model delivers exceptional retrieval performance with a Top-10 accuracy of 0.99, with precision, recall, and F1 scores around 0.98–0.99, regardless of background presence or retrieval method, both the lightweight and heavy models benefit from background removal. Moreover, FAISS consistently outperforms KNN in terms of inference time and CPU efficiency.

Brain Tumour MRI Scans (6.5.1.2)

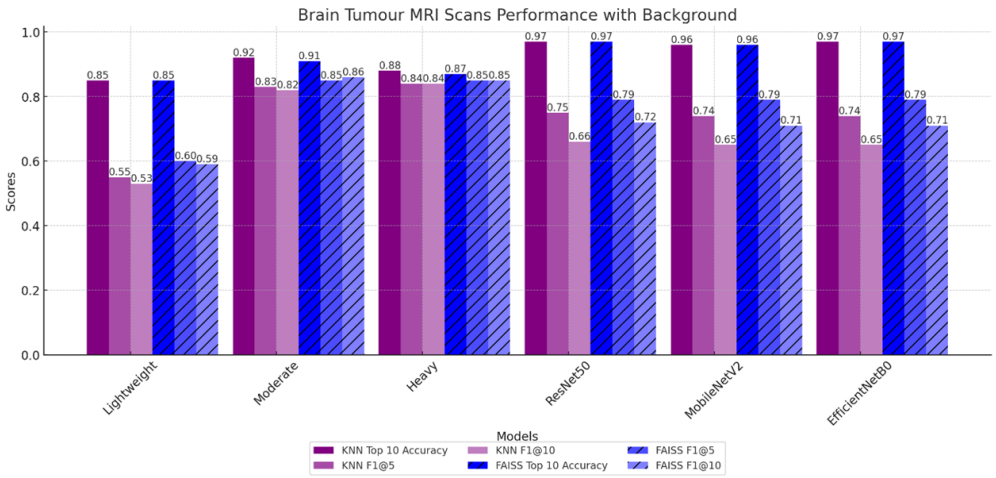

Figure 6.6: Plot showcasing Brain Tumour MRI Scans performance with background present using KNN and FAISS – DSRP Retrieval.

Retrieval Results Figure 6.6 above showcases the retrieval performance for the Brain Tumour MRI Scans dataset under the Domain-Specific Retrieval Pipeline (DSRP) when the background is present. The results indicate that the moderate model achieves an overall balanced performance with a Top-10 accuracy of 0.92 and precision, recall, and F1 scores ranging from 0.80 to 0.86 using both KNN and FAISS. The lightweight model shows a modest performance with a Top-10 accuracy of 0.85 and lower precision and recall, approximately 0.48 to 0.77, with FAISS slightly improving F1 scores to around 0.60. For the heavy model, the Top-10 accuracy is 0.88, with consistent high precision and recall, around 0.83 to 0.87, and only minor differences between KNN and FAISS. The deep learning models such as ResNet50, MobileNetV2, and EfficientNetB0 achieve high Top-10 accuracies, 0.96–0.97, however, at a P@5 level, precision and recall are relatively lower, approximately 0.66 to 0.68, while R@5 remains high around 0.94–0.96. FAISS provides minor improvements in F1 scores for these models overall.

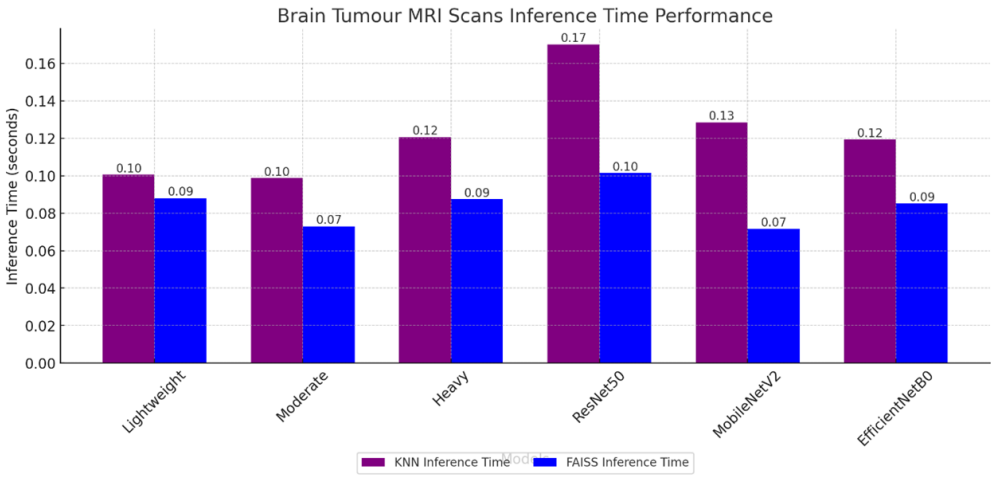

Figure 6.7: Plot showcasing Brain Tumour MRI Scans inference time performance for background conditions using KNN and FAISS – DSRP Retrieval.

Inference Time Performance Figure 6.7 showcases the inference time performance for the Brain Tumour MRI Scans dataset. The lightweight model shows that KNN takes approximately 0.1006 seconds while FAISS improves the time to 0.0881 seconds. The moderate model reduces from 0.0988 seconds with KNN to 0.0730 seconds with FAISS, and for the heavy model, from 0.1207 seconds to 0.0875 seconds. Amongst the deep learning models, ResNet50 is the slowest with KNN around 0.1701 seconds and FAISS about 0.1015 seconds. MobileNetV2 and EfficientNetB0 also show significant improvements in inference time with FAISS.

Hardware Usage The hardware usage for the Brain Tumour MRI Scans dataset was evaluated under background conditions using both KNN and FAISS. Table 6.3 summarizes the CPU and memory usage for each model.

Table 6.3: Hardware Usage Comparison for KNN and FAISS in DSRP Retrieval (Brain Tumour MRI Scans)

Model	Background	KNN		FAISS	
		CPU (%)	Memory (%)	CPU (%)	Memory (%)
Lightweight	Yes	4.1	56.2	3.1	55.3
Moderate	Yes	6.1	50.2	2.1	53.0
Heavy	Yes	4.1	51.2	2.4	46.5
ResNet50	Yes	3.0	50.2	1.2	45.0
MobileNetV2	Yes	3.7	49.2	1.8	44.5
EfficientNetB0	Yes	5.0	52.5	2.0	47.7

The hardware usage results above in Table 6.3 shows that FAISS generally reduces CPU usage compared to KNN across all models. For the lightweight model, CPU usage drops from 4.1% with KNN to 3.1% with FAISS. Similar reductions are observed in the moderate and heavy models, as well as in deep learning models such as ResNet50, MobileNetV2, and EfficientNetB0. Memory usage remains relatively stable, ranging between approximately 44.5% and 56.2%.

These results demonstrate that while the moderate model achieves balanced performance in retrieval metrics, both the lightweight and heavy models benefit from background removal. Additionally, FAISS consistently improves inference time and CPU efficiency compared to KNN.

COVID-19 Scans (6.5.1.3)

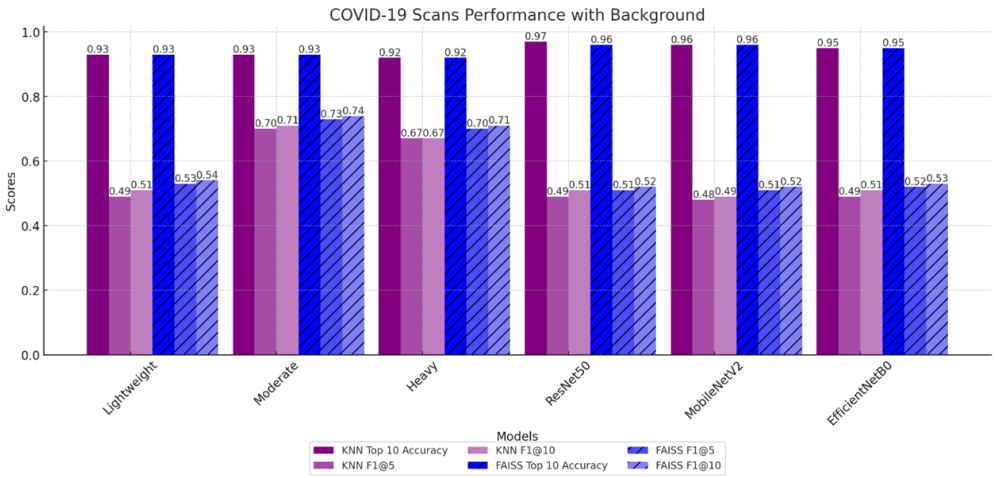

Figure 6.8: Plot showcasing COVID-19 Scans performance with background present using KNN and FAISS – DSRP Retrieval.

Retrieval Results The figure above Figure 6.8 illustrates the retrieval performance for the COVID-19 Scans dataset under the Domain-Specific Retrieval Pipeline (DSRP) when the background is present. The retrieval results shows that the moderate model achieves the most balanced performance with a Top-10 accuracy of 0.93 and relatively good precision, recall, and F1 scores P@5 = 0.62, R@5 = 0.62, F1 of 0.73–0.74. The lightweight model achieves a Top-10 accuracy of 0.93, but with much lower precision P@5 of 0.38, though FAISS improves its F1 scores slightly to around 0.53–0.54. The heavy model achieves a Top-10 accuracy of 0.92 with P@5 around 0.58–0.59 and improved recall and F1 scores approximately 0.70–0.71. For the deep learning models, ResNet50, MobileNetV2, and EfficientNetB0 show high Top-10 accuracy 0.95–0.97, however, their precision at P@5 is lower approximately 0.36–0.37, with modest F1 scores around 0.49–0.52. Overall, FAISS provides minor improvements in F1 scores across all the models.

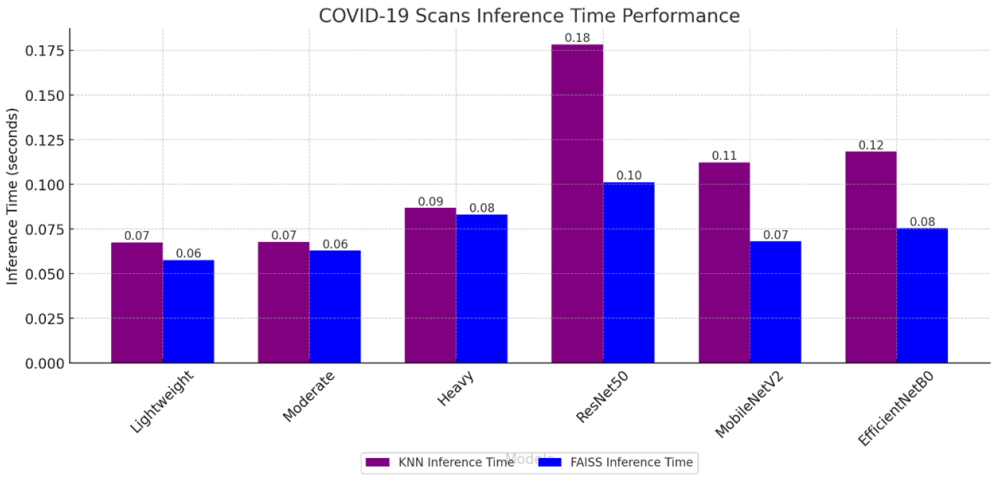

Figure 6.9: Plot showcasing the COVID-19 Scans inference time performance with background conditions present using KNN and FAISS – DSRP Retrieval.

Inference Time Performance The figure above, Figure 6.9 displays the inference time performance for the COVID-19 Scans dataset. The lightweight and moderate models have the fastest inference times, with KNN times around 0.0676–0.0678 seconds and FAISS reducing these times to approximately 0.0577–0.0630 seconds. The heavy model performs slower, with KNN taking 0.0868 seconds and FAISS 0.0831 seconds. Amongst the deep learning models, MobileNetV2 and EfficientNetB0 have a good balance between speed and complexity, MobileNetV2 achieves 0.1123 seconds with KNN, 0.0681 seconds with FAISS, EfficientNetB0 achieves 0.1184 seconds with KNN, 0.0754 seconds with FAISS and ResNet50 achieves 0.1783 seconds with KNN and 0.1013 seconds with FAISS.

Hardware Usage The hardware usage for the COVID-19 Scans dataset was evaluated under background conditions using both KNN and FAISS. Table 6.4 summarizes the CPU and memory usage for each model.

Table 6.4: Hardware Usage Comparison for KNN and FAISS in DSRP Retrieval (COVID-19 Scans)

Model	Background	KNN		FAISS	
		CPU (%)	Memory (%)	CPU (%)	Memory (%)
Lightweight	Yes	2.4	48.4	2.7	48.9
Moderate	Yes	2.4	49.2	2.2	47.9
Heavy	Yes	2.8	44.9	2.2	49.9
ResNet50	Yes	4.8	49.0	2.4	49.1
MobileNetV2	Yes	8.2	48.6	0.7	47.1
EfficientNetB0	Yes	4.6	48.6	1.8	47.2

The hardware usage results in Table 6.4 indicate that the lightweight and moderate models maintain low CPU usage 2.4–2.7% with stable memory consumption around 47.9–49.2% across retrieval methods. The heavy model shows slightly higher CPU usage with KNN around 2.8% that decreases to 2.2% with FAISS. Amongst the deep learning models, ResNet50 and MobileNetV2 experience significant reductions in CPU usage with FAISS, ResNet50 reduces from 4.8% to 2.4%, MobileNetV2 reduces from 8.2% to 0.7% and EfficientNetB0's CPU usage drops from 4.6% to 1.8%. Memory usage remains relatively stable across models. FAISS consistently reduces CPU usage for most models while keeping memory requirements steady overall.

These results demonstrate that while the moderate model achieves balanced retrieval performance for COVID-19 Scans, the lightweight and heavy models benefit from background removal, and FAISS consistently improves inference time and CPU efficiency compared to KNN.

Pokémon (6.5.1.4)

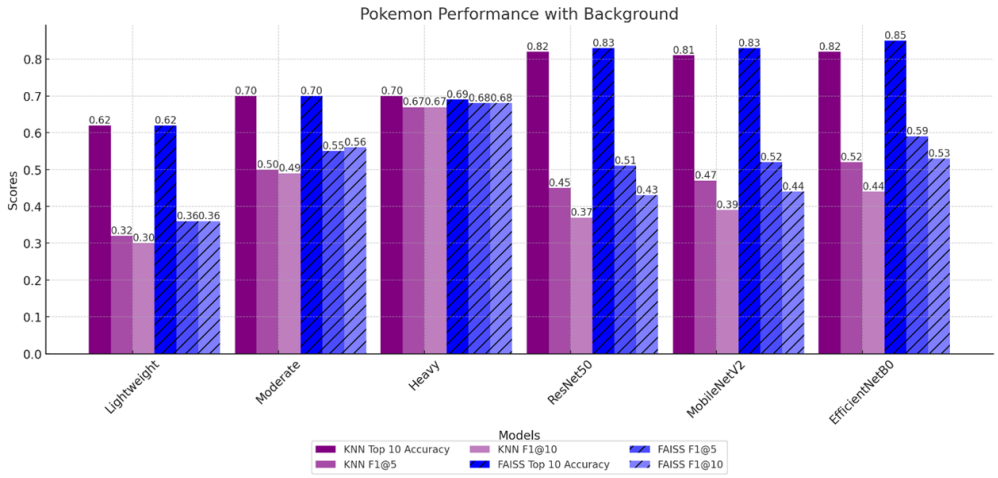

Figure 6.10: Plot showcasing Pokémon performance with background present using KNN and FAISS – DSRP Retrieval.

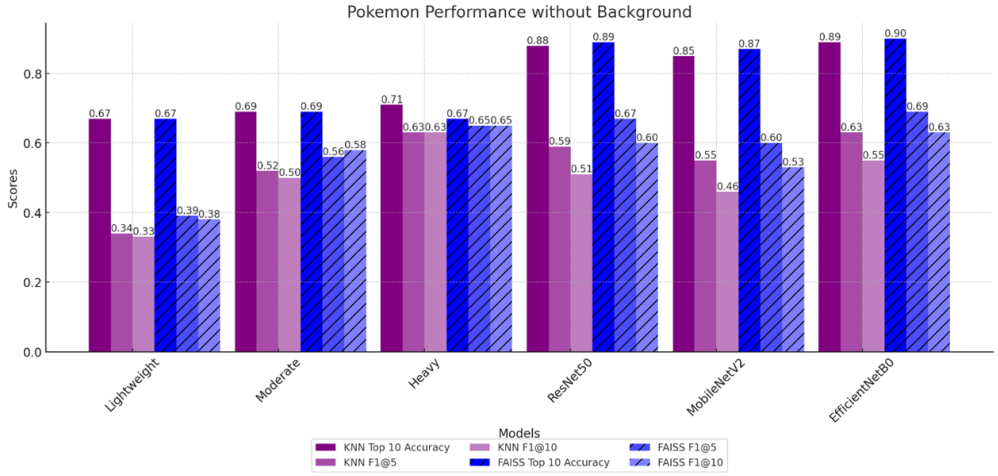

Figure 6.11: Plot showcasing Pokémon performance with no background using KNN and FAISS – DSRP Retrieval.

Retrieval Results The figures above, Figure 6.10 showcases the retrieval performance for the Pokémon dataset under background present conditions, while the figures, Figure 6.11 shows performance with no background present. The results indicate that both the moderate and heavy models perform consistently well in both conditions. The moderate model, achieved a Top-10 accuracy is around 0.69–0.70, with P@5 improving to about 0.48–0.50 without background and a stable F1 score of roughly 0.56–0.58. The heavy model achieves a Top-10 accuracy of approximately 0.70, with better precision and recall, P@5 0.66 and F1 0.67–0.68, consistently across both

retrieval methods. The lightweight model performed the worst, with a Top-10 accuracy ranging from 0.62 to 0.67 and a P@5 score of around 0.26–0.30, though FAISS slightly improves F1 scores to approximately 0.36–0.39. Amongst the deep learning models, ResNet50 reaches an accuracy of 0.82–0.89, especially without background present, where precision is higher, P@5 0.51–0.55, and F1 scores range from 0.60–0.67. The MobileNetV2 model performs better without background present with a Top-10 accuracy of 0.85–0.87 and F1 scores around 0.60, while the EfficientNetB0 model achieves the best overall performance with Top-10 accuracy of 0.89–0.90, and strong precision and F1 scores, P@5 0.55–0.58, F1 0.63–0.69. FAISS provides minor improvements in F1 scores for all models overall.

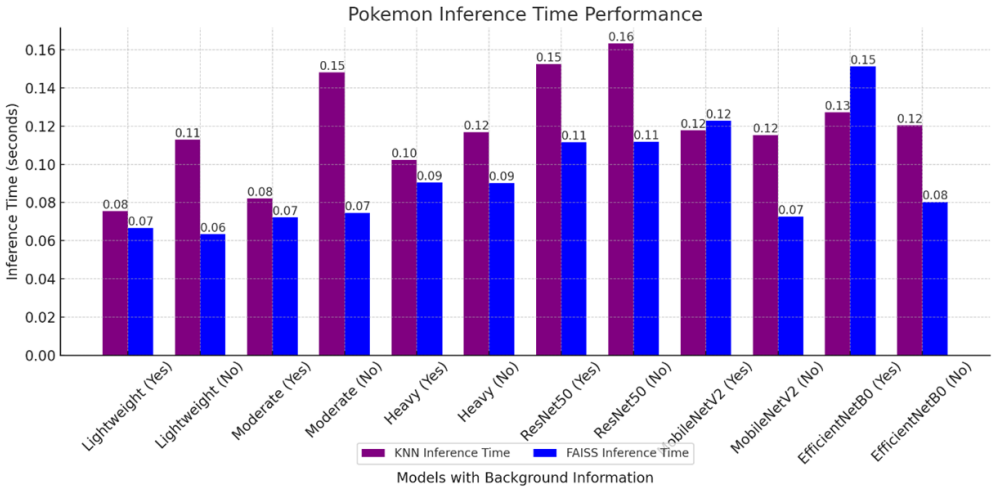

Figure 6.12: Plot showcasing the Pokémon inference time performance for background and no background conditions using KNN and FAISS – DSRP Retrieval.

Inference Time Performance The figure above, Figure 6.12 showcases the inference time performance for the Pokémon dataset. The inference times indicate that FAISS consistently outperforms KNN, particularly in no background present scenarios. The lightweight model shows an increase in KNN inference time without background from 0.0754 to 0.1128 seconds, while achieving better times with FAISS, 0.0667 seconds with background and 0.0633 seconds without. Similar trends are observed for the moderate and heavy models, with MobileNetV2 being the fastest among deep learning models, and EfficientNetB0 also benefiting significantly from FAISS.

Hardware Usage The hardware usage for the Pokémon dataset was evaluated under background conditions using both KNN and FAISS. Table 6.5 summarizes the CPU and memory usage for each model.

Table 6.5: Hardware Usage Comparison for KNN and FAISS in DSRP Retrieval (Pokémon)

Model	Background	KNN		FAISS	
		CPU (%)	Memory (%)	CPU (%)	Memory (%)
Lightweight	Yes	4.4	47.1	1.4	52.9
Lightweight	No	4.3	47.0	0.9	46.6
Moderate	Yes	3.7	47.3	1.5	53.2
Moderate	No	6.1	49.0	2.0	49.5
Heavy	Yes	4.9	48.9	1.2	53.5
Heavy	No	3.7	53.1	2.2	48.4
ResNet50	Yes	4.2	51.6	1.9	53.1
ResNet50	No	4.4	51.2	2.0	48.3
MobileNetV2	Yes	2.4	53.3	2.6	44.0
MobileNetV2	No	2.8	49.8	2.7	47.7
EfficientNetB0	Yes	5.8	53.0	2.5	44.8
EfficientNetB0	No	10.9	41.0	3.1	47.7

The hardware usage results above in Table 6.5 showcases that, on average, FAISS reduces CPU usage across models under background conditions. For the lightweight model, CPU usage decreases from 4.4% with KNN to 1.4% with FAISS, while memory usage remains stable around 47–53%. The moderate model sees CPU usage drop from 3.7% using KNN to 1.5–2.0% using FAISS, and similar trends are observed for the heavy model and deep learning models, ResNet50, MobileNetV2, EfficientNetB0. FAISS consistently improves CPU efficiency while memory consumption remains nearly identical across models overall.

These results demonstrate that while the moderate model achieves stable retrieval performance for Pokémon, both the lightweight and heavy models benefit from background removal. Additionally, FAISS consistently enhances inference time and reduces CPU usage compared to KNN overall.

Technical Summary: Domain-Specific Retrieval Pipeline (DSRP)

The experimental evaluation in the above section was performed using the Domain-Specific Retrieval Pipeline (DSRP) on multiple datasets, including the Leaf Disease, Brain Tumour MRI Scans, COVID-19 Scans, and Pokémon images. The evaluation compared various models, from custom lightweight, moderate, and heavy architectures to deep learning models ResNet50, MobileNetV2, and EfficientNetB0, using two retrieval methods, KNN and FAISS.

Inference Time Performance

Across all datasets, FAISS consistently reduced inference times relative to KNN.

- **Leaf Disease:** For the lightweight model, KNN required approximately 0.0806 seconds with background, 0.0807 seconds without background, which was reduced to 0.0665 seconds with background and 0.0711 seconds without background using FAISS. Similar trends were observed for the moderate, heavy, and deep learning models.
- **Brain Tumour MRI Scans:** The lightweight model's inference time improved from about 0.1006 seconds using KNN to 0.0881 seconds using FAISS, with similar trends observed in the moderate and heavy models.
- **COVID-19 Scans:** Lightweight and moderate models achieved inference times around 0.0676–0.0678 seconds with KNN, which were further reduced with FAISS, however, the heavy model and deep learning models, especially ResNet50, exhibited longer inference times.
- **Pokémon:** FAISS outperformed KNN, particularly in no-background scenarios, with deep learning models like MobileNetV2 and EfficientNetB0 achieving a favorable balance between speed and complexity.

Hardware Utilization

The evaluation of CPU and memory usage revealed that FAISS accelerates inference while reducing CPU load across all models, with memory usage remaining relatively stable.

- **Leaf Disease:** For the lightweight model, the CPU usage decreased dramatically from 4.8% with KNN to 0.1% with FAISS when background was present, similar trends were observed in the moderate, heavy, and deep learning models. The memory usage varied between 47.4% and 57.9% overall.

- **Brain Tumour MRI Scans:** The lightweight model's CPU usage dropped from 4.1% with KNN to 3.1% with FAISS, while memory consumption remained relatively stable.
- **COVID-19 Scans:** The deep learning models like ResNet50 and MobileNetV2 experienced significant CPU reductions, ResNet50 from 4.8% to 2.4% and MobileNetV2 from 8.2% to 0.7%, with nearly constant memory usage.
- **Pokémon:** FAISS consistently reduced the CPU usage under both background present and no-background present conditions, with memory usage remaining within the approximate range of 47% to 53%.

Retrieval Performance

Retrieval metrics, including Top-10 accuracy, precision, recall, and F1 scores, were analyzed across the datasets.

- **Leaf Disease:** The moderate model delivered exceptional performance with a Top-10 accuracy of 0.99 and precision, recall, and F1 scores around 0.98–0.99 regardless of background. The lightweight model achieved 0.98 accuracy with background and 0.96 without, while the heavy model improved from 0.94 to 0.99 without background. The deeep learning models, ResNet50, MobileNetV2, EfficientNetB0, consistently reached around 0.99 accuracy overall.
- **Brain Tumour MRI Scans:** The lightweight model's retrieval performance improved with FAISS, though precision metrics were lower compared to the deeper models. While the lightweight model maintained similar Top-10 accuracy with or without FAISS, the deeper models benefited more in terms of retrieval quality overall.
- **COVID-19 Scans:** The moderate model achieved balanced performance with a Top-10 accuracy of approximately 0.93 and strong precision, recall, and F1 scores, P@5 around 0.62 and F1 between 0.73 and 0.74. The lightweight model, despite similar accuracy, exhibited lower precision. The deep learning models showed high Top-10 accuracy of 0.95–0.97 but modest precision.
- **Pokémon:** The moderate and heavy models both maintained consistent performance across background present and no-background present conditions. The moderate model reached a Top-10 accuracy near 0.70, while the heavy model demonstrated improved precision and F1 scores. Amongst the deep learning models, EfficientNetB0 achieved the highest performance with Top-10 accuracy between 0.89 and 0.90.

Overall Findings

The experimental results demonstrate that:

- **Efficiency Gains:** FAISS consistently outperforms KNN by reducing inference times and lowering CPU usage, while memory usage remains stable.
- **Model Trade-offs:** The moderate model generally provides balanced performance across retrieval metrics, whereas both lightweight and heavy models benefit significantly from background removal. The deep learning models, although achieving high accuracy, may incur longer inference times, notably in the case of ResNet50.
- **DSRP Effectiveness:** The Domain-Specific Retrieval Pipeline (DSRP) effectively integrates diverse models and retrieval methods to optimize both performance and hardware utilization across varied datasets.

These findings offer valuable insights into the design and optimization of image retrieval tasks, demonstrating that incorporating FAISS within the DSRP framework enhances computational efficiency and retrieval performance overall.

6.6.2 Pre-trained Feature Extraction Pipeline (PMRP)

Leaf Disease (6.5.2.1)

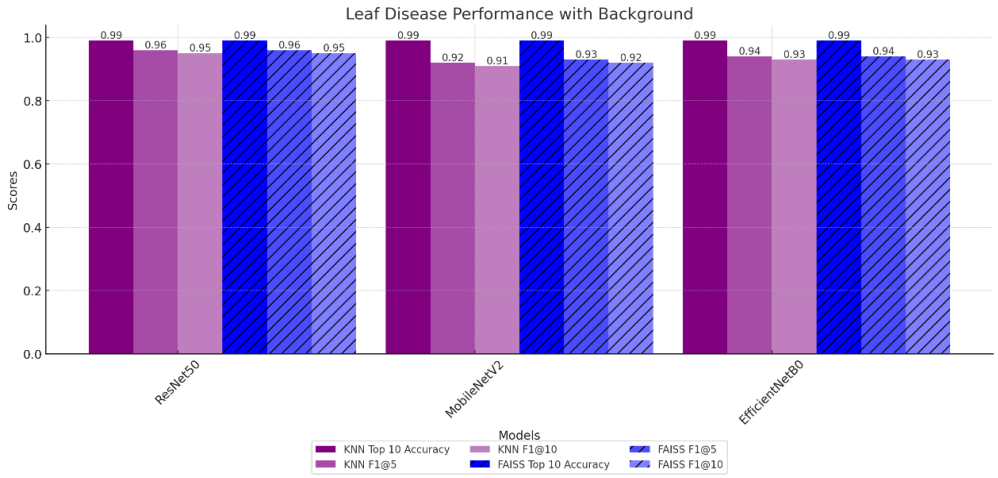

Figure 6.13: Plot showcasing Leaf Disease performance with background present using KNN and FAISS – PMRP Retrieval.

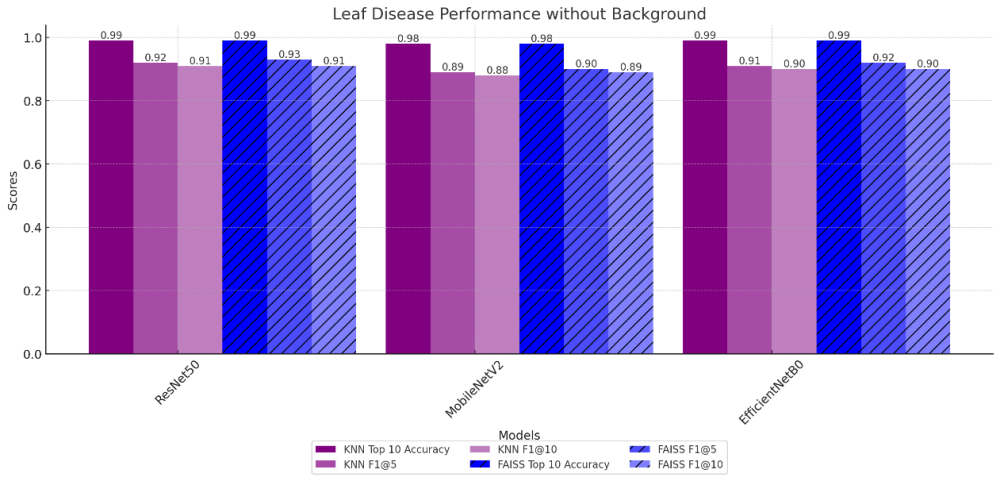

Figure 6.14: Plot showcasing Leaf Disease performance with no background using KNN and FAISS – PMRP Retrieval.

Retrieval Results The figures above, Figures 6.13 and 6.14 showcases the retrieval performance for the Leaf Disease dataset under the PMRP retrieval pipeline. The results show that for ResNet50, MobileNetV2, and EfficientNetB0, the Top-10 accuracies are very high, 0.98–0.99. The ResNet50 model maintains a Top-10 accuracy of 0.99 with very strong precision and recall, P@5 of 0.94 with background and 0.90 without, and its F1 score remains high at approximately 0.96 regardless

of background. The MobileNetV2 model exhibits some degradation without background, P@5 decreases from 0.89 to 0.86 with KNN and from 0.91 to 0.87 with FAISS, with FAISS slightly improving F1 scores, 0.93 with background versus 0.90 without. The EfficientNetB0 model behaves similarly, with a slight drop in precision and recall, P@5 from 0.92 to 0.88 with KNN, although FAISS provides a marginal boost, F1 around 0.94 with background and 0.92 without.

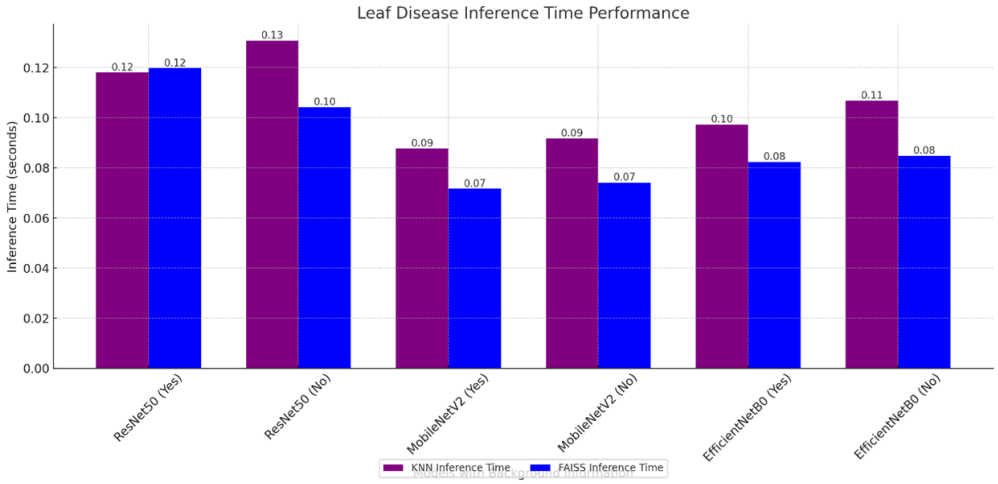

Figure 6.15: Plot showcasing the Leaf Disease inference time performance for background and no background conditions using KNN and FAISS – PMRP Retrieval.

Inference Time Performance The figure above, Figure 6.15 showcases that for most models, FAISS slightly improves inference efficiency compared to KNN, especially without background present. The ResNet50 models inference time improves from 0.1197 seconds KNN with background present to 0.1041 seconds FAISS without background present. Similar trends are observed for MobileNetV2 and EfficientNetB0.

Hardware Usage The hardware usage for the Leaf Disease dataset under PMRP was evaluated for ResNet50, MobileNetV2, and EfficientNetB0. Table 6.6 summarizes the CPU and memory usage for each model under background conditions, comparing KNN and FAISS.

Table 6.6: Hardware Usage Comparison for KNN and FAISS in PMRP Retrieval (Leaf Disease)

Model	Background	KNN		FAISS	
		CPU (%)	Memory (%)	CPU (%)	Memory (%)
ResNet50	Yes	13.0	48.4	12.6	52.2
ResNet50	No	10.7	54.2	12.6	52.5
MobileNetV2	Yes	14.7	44.7	12.1	54.8
MobileNetV2	No	13.6	51.9	10.2	52.1
EfficientNetB0	Yes	12.9	43.3	11.7	50.5
EfficientNetB0	No	14.2	48.9	11.8	48.4

The hardware usage results above in Table 6.6 showcases that FAISS generally reduces CPU usage compared to KNN. The MobileNetV2 model shows a reduction from 14.7% using KNN with background, to 12.1% using FAISS with background, and similar trends are observed for EfficientNetB0. The ResNet50 model shows comparable CPU usage with background, though without background FAISS slightly increases usage compared to KNN. Memory usage remains relatively stable across models.

These results demonstrate that under the PMRP, while retrieval performance for the Leaf Disease dataset remains high for deep learning models, FAISS consistently improves inference time and reduces CPU usage compared to KNN, with only minor fluctuations in memory usage overall.

Brain Tumour MRI Scans (6.5.2.2)

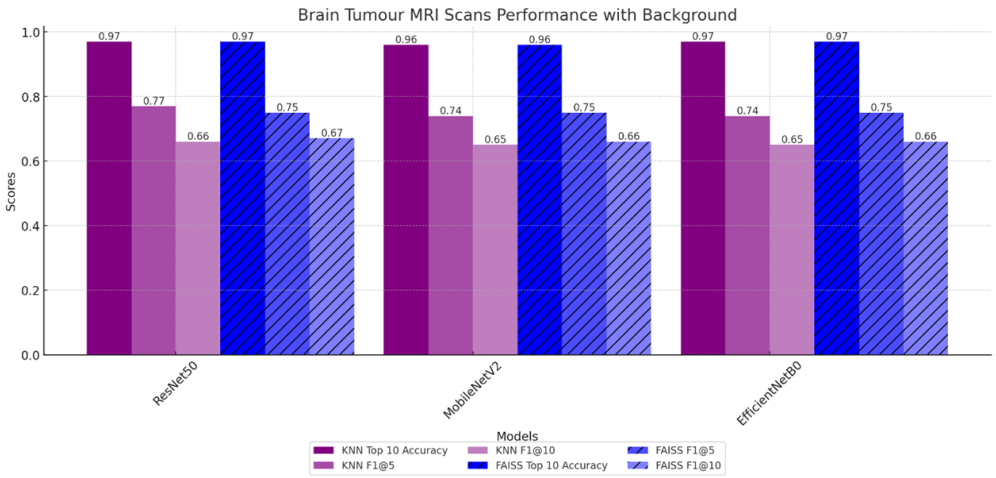

Figure 6.16: Plot showcasing Brain Tumour MRI Scans performance with background present using KNN and FAISS – PMRP Retrieval.

Retrieval Results The figure above, Figure 6.16 showcases the retrieval performance for the Brain Tumour MRI Scans dataset under the PMRP retrieval pipeline when the background is present. The results reveal that ResNet50, MobileNetV2, and EfficientNetB0 achieve high Top-10 accuracies, approximately 0.96–0.97, across both KNN and FAISS. However, their precision and recall metrics are noticeably lower. For the ResNet50 model, P@5 and R@5 are 0.67 and 0.95, respectively, with FAISS yielding F1 scores around 0.77 with KNN and 0.75 with FAISS. The MobileNetV2 model shows similar performance, with P@5 approximately 0.66–0.67 and F1 scores close to 0.74–0.75. The EfficientNetB0 model follows this pattern, with a Top-10 accuracy of about 0.97, and slight improvements by FAISS, P@5 values of 0.67 and F1 scores around 0.75. FAISS provides minor improvements in F1 scores across these models overall.

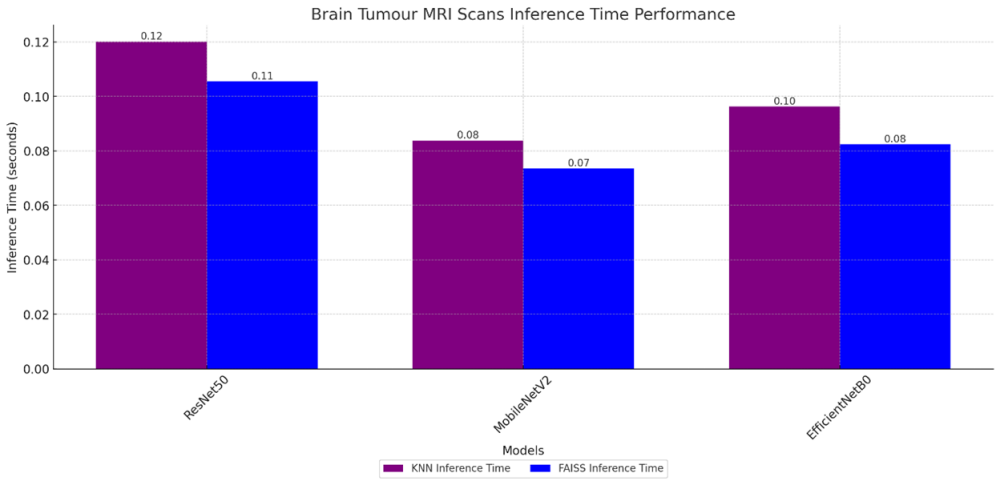

Figure 6.17: Plot showcasing Brain Tumour MRI Scans inference time performance with background conditions present using KNN and FAISS – PMRP Retrieval.

Inference Time Performance The figure above, Figure 6.17 showcases the inference time performance. For instance, ResNet50's inference time decreases from 0.1201 seconds with KNN to 0.1055 seconds with FAISS, MobileNetV2 improves from 0.0838 seconds using KNN to 0.0736 seconds using FAISS, and EfficientNetB0's time is reduced from 0.0963 seconds to 0.0824 seconds when using FAISS.

Hardware Usage The hardware usage for the Brain Tumour MRI Scans dataset under PMRP was evaluated using both KNN and FAISS. Table 6.7 summarizes the CPU and memory usage for each model under background conditions.

Table 6.7: Hardware Usage Comparison for KNN and FAISS in PMRP Retrieval (Brain Tumour MRI Scans)

Model	Background	KNN		FAISS	
		CPU (%)	Memory (%)	CPU (%)	Memory (%)
ResNet50	Yes	11.6	44.9	11.3	47.5
MobileNetV2	Yes	12.7	49.9	11.3	49.2
EfficientNetB0	Yes	12.5	51.7	11.5	49.9

The hardware usage results above in Table 6.7 showcases that FAISS slightly reduces CPU usage compared to KNN across all the models. The ResNet50 model CPU usage decreases from 11.6% with KNN to 11.3% with FAISS, the MobileNetV2 model drops from 12.7% to 11.3%, and the

EfficientNetB0 model decreases from 12.5% to 11.5%. The memory usage shows only minor fluctuations, remaining generally comparable between the methods.

These results demonstrate that under the PMRP retrieval pipeline, the deep learning models maintain high retrieval accuracy with FAISS yielding modest improvements in inference time and CPU efficiency, while memory consumption remains stable overall.

COVID-19 Scans (6.5.2.3)

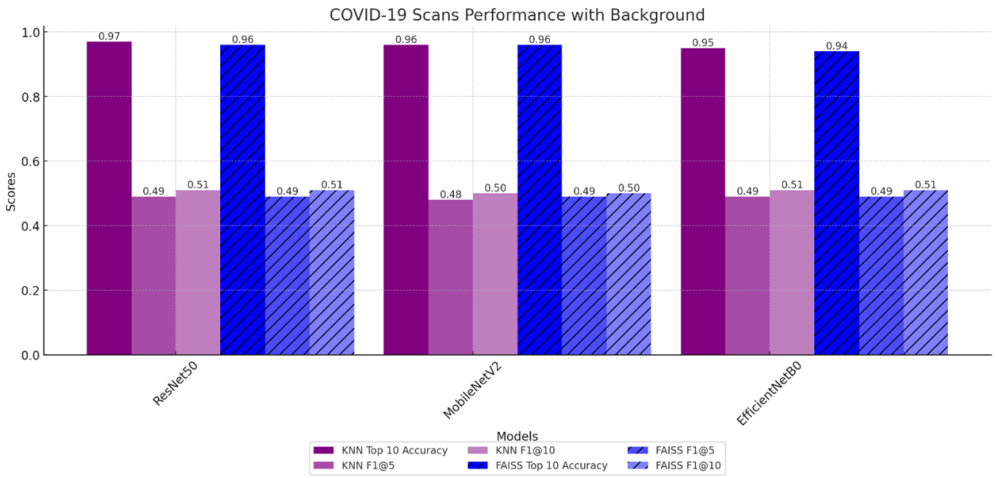

Figure 6.18: Plot showcasing COVID-19 Scans performance with background present using KNN and FAISS – PMRP Retrieval.

Retrieval Results The figure above, Figure 6.18 showcases the retrieval performance for the COVID-19 Scans dataset under the PMRP retrieval pipeline when the background is present. The results indicate that ResNet50, MobileNetV2, and EfficientNetB0 all achieve high Top-10 accuracy, approximately 0.94–0.97, with both KNN and FAISS. However, their precision and recall scores are considerably lower. The ResNet50 model achieves a P@5 of only 0.36 with KNN, with a higher R@5 of 0.87, while FAISS provides a slight improvement in F1 scores increasing from 0.49 to 0.51. The MobileNetV2 model shows similar performance, with P@5 values of about 0.36 for KNN that improve marginally when using FAISS, resulting in F1 scores around 0.49–0.50. The EfficientNetB0 model exhibits a P@5 of 0.37, with its F1 score remaining close, around 0.49–0.51, when using FAISS.

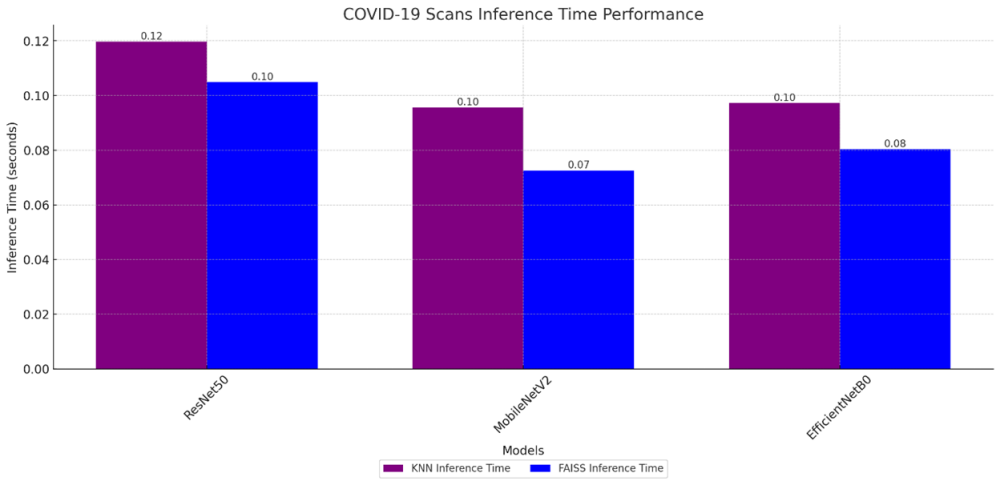

Figure 6.19: Plot showcasing the COVID-19 Scans inference time performance with background conditions present using KNN and FAISS – PMRP Retrieval.

Inference Time Performance The figure above, Figure 6.19 showcases that FAISS improves inference time across all models. The ResNet50 model inference time decreases from 0.1197 seconds with KNN to 0.1049 seconds with FAISS, the MobileNetV2 models inference time drops from 0.0956 seconds using KNN to 0.0726 seconds using FAISS, and the EfficientNetB0 model inference time reduces from 0.0973 seconds to 0.0804 seconds with FAISS.

Hardware Usage The hardware usage for the COVID-19 Scans dataset under PMRP was evaluated for ResNet50, MobileNetV2, and EfficientNetB0. Table 6.8 summarizes the CPU and memory usage for each model under background conditions.

Table 6.8: Hardware Usage Comparison for KNN and FAISS in PMRP Retrieval (COVID-19 Scans)

Model	Background	KNN		FAISS	
		CPU (%)	Memory (%)	CPU (%)	Memory (%)
ResNet50	Yes	10.3	49.0	14.6	49.1
MobileNetV2	Yes	13.5	48.5	10.3	48.2
EfficientNetB0	Yes	13.5	47.2	9.7	47.7

The hardware usage results above in Table 6.8 showcases that for the COVID-19 Scans dataset, ResNet50 experiences an increase in CPU usage from 10.3% with KNN to 14.6% with FAISS. The MobileNetV2 model CPU usage decreases from 13.5% to 10.3%, and the EfficientNetB0 model

CPU usage drops from 13.5% to 9.7%. The memory usage remains relatively stable across models, typically ranging between 47% and 49%.

These results demonstrate that under the PMRP, while deep learning models achieve high Top-10 accuracy for COVID-19 Scans, their precision and recall scores are lower. FAISS improves inference time and CPU efficiency for MobileNetV2 and EfficientNetB0, although it increases CPU usage for ResNet50 overall.

Pokémon (6.5.2.4)

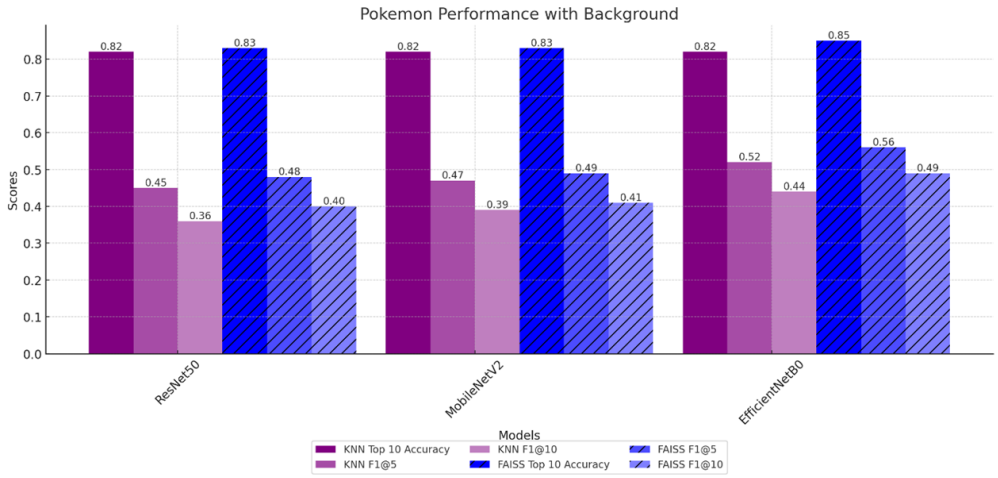

Figure 6.20: Plot showcasing Pokémon performance with background present using KNN and FAISS – PMRP Retrieval.

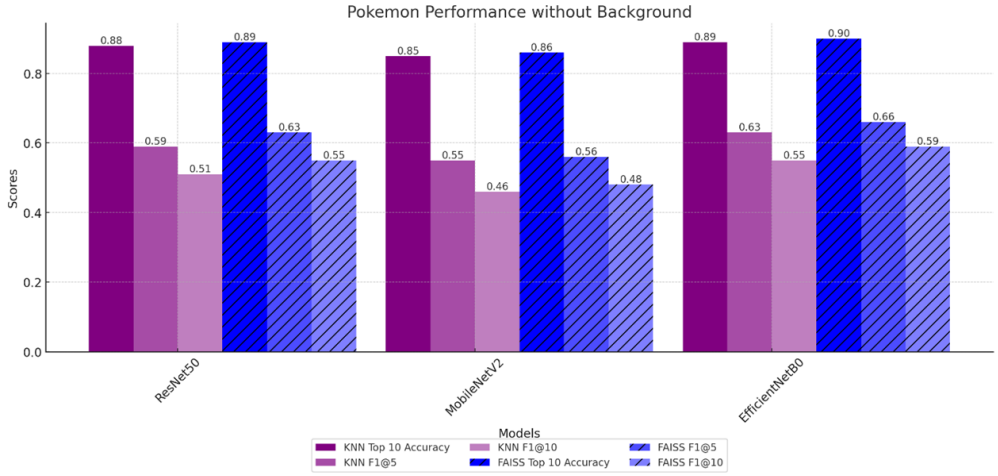

Figure 6.21: Plot showcasing Pokémon performance with no background using KNN and FAISS – PMRP Retrieval.

Retrieval Results The figures above, Figures 6.20 and 6.21 showcases the retrieval performance for the Pokémon dataset under the PMRP retrieval pipeline. The results indicate that for ResNet50, MobileNetV2, and EfficientNetB0, the Top-10 accuracies range from approximately 0.82–0.85 with background and 0.88–0.90 without background. When the background is present, the precision and recall scores are relatively low, the ResNet50 model achieves a P@5 of 0.35 and an F1@5 of 0.45. These metrics improve when the background is removed, with P@5 increasing to 0.51–0.55 and F1 scores reaching around 0.63 when using FAISS. The MobileNetV2 model also achieves an

improvement in precision from 0.38 to 0.46–0.48 without background, while the EfficientNetB0 model outperforms the others with a P@5 of 0.44 with background and 0.55 without, and its F1 scores increase from 0.52 to 0.66 with FAISS, indicating a well-balanced precision-recall trade-off.

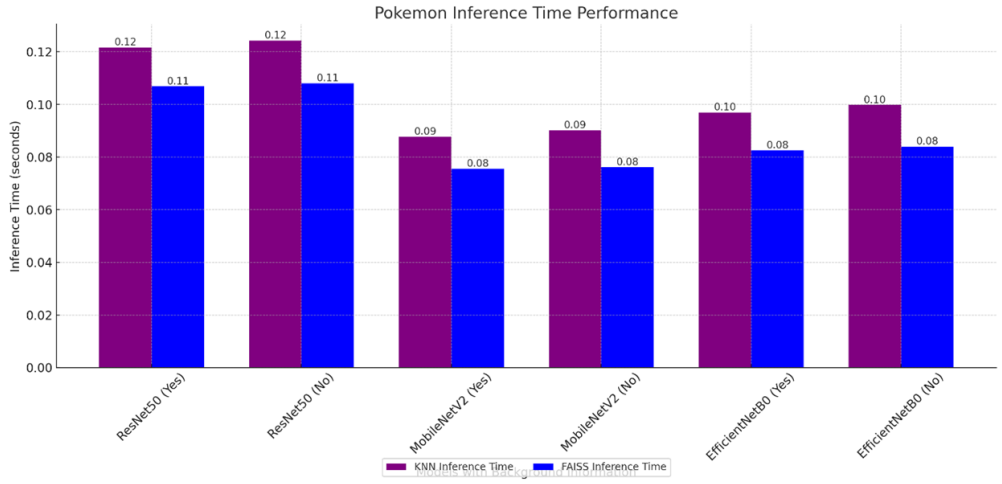

Figure 6.22: Plot showcasing the Pokémon inference time performance for background and no background conditions using KNN and FAISS – PMRP Retrieval.

Inference Time Performance The figure above, Figure 6.22 showcases that FAISS improves inference time across models. The ResNet50 model inference time is reduced from 0.1216 seconds using KNN to 0.1068 seconds using FAISS with background, and from 0.1242 to 0.1080 seconds without background. The MobileNetV2 and EfficientNetB0 models also show improvements, the MobileNetV2 model decreases from 0.0876 seconds using KNN to 0.0755 seconds using FAISS with background, and the EfficientNetB0 model from 0.0969 seconds to 0.0825 seconds with background, with similar trends observed in the absence of the background features.

Hardware Usage The hardware usage for the Pokémon dataset under PMRP was evaluated for ResNet50, MobileNetV2, and EfficientNetB0. Table 6.9 summarizes the CPU and memory usage for each model under background conditions, comparing KNN and FAISS.

Table 6.9: Hardware Usage Comparison for KNN and FAISS in PMRP Retrieval (Pokémon)

Model	Background	KNN		FAISS	
		CPU (%)	Memory (%)	CPU (%)	Memory (%)
ResNet50	Yes	13.4	47.6	11.3	48.9
ResNet50	No	12.3	47.8	10.1	48.0
MobileNetV2	Yes	13.0	48.2	11.7	49.3
MobileNetV2	No	12.5	48.5	12.0	49.9
EfficientNetB0	Yes	12.8	48.8	10.6	48.7
EfficientNetB0	No	12.8	48.2	11.5	50.2

The hardware usage above in Table 6.9 showcases that for the ResNet50 model, CPU usage decreases from 13.4% with KNN to 11.3% with FAISS when the background is present, and from 12.3% to 10.1% without background. The MobileNetV2 model achieves a CPU usage drop from 13.0% to 11.7% with background, while remaining nearly steady from 12.5% to 12.0% without background. The EfficientNetB0 model shows a reduction from 12.8% using KNN to 10.6% using FAISS with background, and from 12.8% to 11.5% without background. The memory usage across models remains relatively stable, generally ranging between 47% and 50%. These findings suggest that FAISS generally offers CPU efficiency advantages, especially when backgrounds are present.

These results indicate that under PMRP, while deep learning models achieve high Top-10 accuracy for Pokémon retrieval, their precision and recall scores are relatively lower. FAISS slightly improves inference time and CPU efficiency, particularly for MobileNetV2 and EfficientNetB0, and enhances F1 scores modestly overall.

Technical Summary: Pre-trained Feature Extraction Pipeline (PMRP)

The experimental evaluation in the section above was performed using the Pre-trained Feature Extraction Pipeline (PMRP) on multiple datasets, including Leaf Disease, Brain Tumour MRI Scans, COVID-19 Scans, and Pokémon images. The results compared deep learning models ResNet50, MobileNetV2, and EfficientNetB0, using two retrieval methods, KNN and FAISS.

Retrieval Performance

- **Leaf Disease:** Figures 6.13 and 6.14 indicate that all models achieve very high Top-10 accuracies (0.98–0.99). ResNet50 maintains a Top-10 accuracy of 0.99 with strong precision and recall (P@5 of 0.94 with background and 0.90 without) and an F1 score of approximately 0.96. MobileNetV2 shows slight degradation without background (P@5 decreases from 0.89 to 0.86 with KNN and from 0.91 to 0.87 with FAISS), with FAISS marginally improving F1 scores (0.93 with background versus 0.90 without). EfficientNetB0 follows a similar pattern, with a small drop in P@5 from 0.92 to 0.88 with KNN and a marginal F1 boost provided by FAISS (around 0.94 with background and 0.92 without).

- **COVID-19 Scans:** Figure 6.18 shows that all models achieve high Top-10 accuracies (approximately 0.94–0.97) with both KNN and FAISS; however, precision and recall remain lower. For instance, ResNet50 obtains a P@5 of 0.36 with KNN (R@5 of 0.87), with FAISS slightly improving the F1 score (from 0.49 to 0.51). Similar trends are noted for MobileNetV2 and EfficientNetB0.

- **Pokémon:** Figures 6.20 and 6.21 reveal that for Pokémon, Top-10 accuracies range from approximately 0.82–0.85 with background and 0.88–0.90 without background. With background, ResNet50 achieves a P@5 of 0.35 and an F1@5 of 0.45, which improve when the background is removed (P@5 increases to 0.51–0.55 and F1 to around 0.63 with FAISS). MobileNetV2 shows an improvement in precision from 0.38 to 0.46–0.48 without background, while EfficientNetB0 attains a P@5 of 0.44 with background and 0.55 without, with its F1 score increasing from 0.52 to 0.66 with FAISS.

Inference Time Performance

Across all datasets, FAISS consistently reduced inference times relative to KNN.

- **Leaf Disease:** Figures 6.13–6.15 show that for the Leaf Disease dataset, FAISS improves inference efficiency. For example, ResNet50's inference time improves from 0.1197 seconds (KNN, with background) to 0.1041 seconds (FAISS, without background), with similar trends observed for MobileNetV2 and EfficientNetB0.

- **Brain Tumour MRI Scans:** For this dataset, ResNet50's inference time decreases from 0.1201 seconds (KNN) to 0.1055 seconds (FAISS), MobileNetV2 from 0.0838 seconds to 0.0736 seconds, and EfficientNetB0 from 0.0963 seconds to 0.0824 seconds.

- **COVID-19 Scans:** Figure 6.19 illustrates that FAISS reduces inference time across all models, with ResNet50 improving from 0.1197 seconds (KNN) to 0.1049 seconds (FAISS), MobileNetV2 from 0.0956 seconds to 0.0726 seconds, and EfficientNetB0 from 0.0973 seconds to 0.0804 seconds.

- **Pokémon:** Figures 6.20–6.22 show that for Pokémon, ResNet50's inference time is reduced from 0.1216 seconds (KNN) to 0.1068 seconds (FAISS) with background (and from 0.1242 to 0.1080 seconds without), while MobileNetV2 and EfficientNetB0 display similar improvements.

Hardware Utilization

Evaluation of CPU and memory usage indicates that FAISS generally offers modest gains in CPU efficiency while memory consumption remains stable.

- **Leaf Disease:** As shown in Table 6.6, for ResNet50, CPU usage is 13.0% (KNN) versus 12.6% (FAISS) with background, and 10.7% versus 12.6% without background. MobileNetV2 drops from 14.7% to 12.1% (with background) and from 13.6% to 10.2% (without), while EfficientNetB0 decreases from 12.9% to 11.7% (with background) and from 14.2% to 11.8% (without). Memory usage remains within a similar range (approximately 43%–57%).

- **Brain Tumour MRI Scans:** Table 6.7 indicates that for ResNet50, CPU usage decreases from 11.6% (KNN) to 11.3% (FAISS) and memory usage shifts slightly from 44.9% to 47.5%. Similar improvements are observed for MobileNetV2 and EfficientNetB0.

- **COVID-19 Scans:** According to Table 6.8, ResNet50's CPU usage increases from 10.3% (KNN) to 14.6% (FAISS), while MobileNetV2 and EfficientNetB0 see decreases from 13.5% to 10.3% and 13.5% to 9.7%, respectively; memory usage remains nearly constant (around 47%–49%).

- **Pokémon:** Table 6.9 shows that for ResNet50, CPU usage decreases from 13.4% (KNN) to 11.3% (FAISS) with background (and from 12.3% to 10.1% without), MobileNetV2 drops from 13.0% to 11.7% with background (remaining nearly steady without), and EfficientNetB0 decreases from 12.8% to 10.6% with background (and from 12.8% to 11.5% without). Memory usage generally ranges between 47% and 50%.

Overall Findings

Overall, the PMRP results indicate that:

- **High Retrieval Accuracy:** Deep learning models maintain high Top-10 accuracy across datasets.
- **Inference Time Improvements:** FAISS consistently improves inference time across all datasets, with MobileNetV2 and EfficientNetB0 exhibiting the most notable gains.
- **CPU Efficiency:** FAISS generally reduces CPU usage, although in the COVID-19 Scans dataset, ResNet50 shows an increase. Memory usage remains stable across methods.
- **Precision and Recall Trade-offs:** While retrieval performance (in terms of Top-10 accuracy) is high, precision and recall are lower for COVID-19 Scans and Pokémon, with FAISS providing modest improvements in F1 scores.

In summary, the Pre-trained Feature Extraction Pipeline (PMRP) effectively leverages deep learning models to achieve high retrieval accuracy, with FAISS offering incremental benefits in computational efficiency and retrieval performance.

6.7 Visual Results

6.7.1 Retrieval Results

Before presenting the figures, it is important to describe the query image selection process. For this experiment, two images were randomly selected from the test subset of the Leaf Disease, Brain Tumour MRI Scans, COVID-19 Scans and Pokemon datasets, ensuring that the model had no prior exposure to these images during training.

Furthermore, for any selected image that also exists in the dataset with background removed, i.e. the Leaf Disease and Pokemon datasets, its corresponding background removed version was included in the retrieval results to evaluate the impact of background removal.

In each figure, the query images are displayed alongside their 10 closest retrieval neighbors as determined by both the KNN and FAISS methods. Each neighbor is annotated with a color indication outline, a green outline denotes a correct class match (i.e., classification), while a blue outline indicates an incorrect match.

The images are arranged in corresponding order based on the retrieval accuracy metric, which reflects the distance of the match for both KNN and FAISS. The query and retrieved images classes are also displayed above each for a more clear indication.

It is also important to note that there are two retrieval pipelines, the Domain-Specific Retrieval Pipeline (DSRP) and the Pre-trained Feature Extraction Pipeline (PMRP). However, only the results for the DSRP pipeline are shown below. The PMRP pipeline exhibits extremely high accuracies, resulting in all retrievals being correct. By highlighting the DSRP findings, we can visually inspect instances of misclassification and gain deeper insights into the performance of the domain-specific retrieval approach.

Leaf Disease (6.6.1.1)

In this section, it shows the retrieval performance for the Leaf Disease dataset with background present.

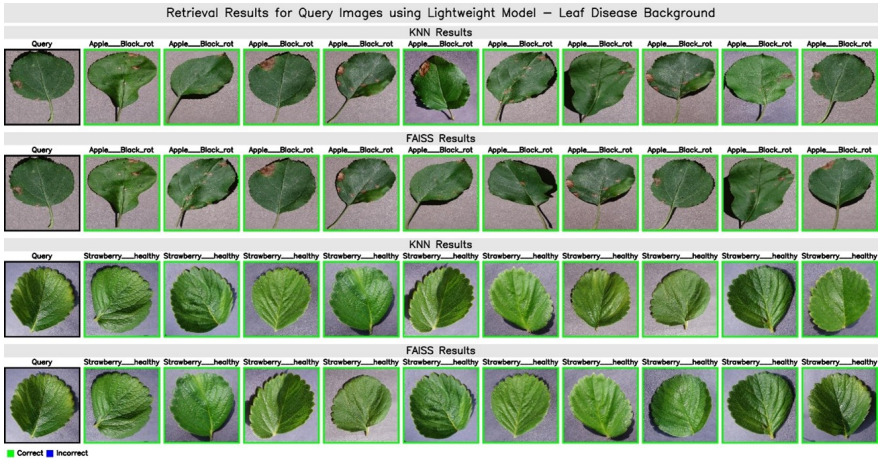

Figure 6.23: Retrieval Results for Query Images using the Lightweight Model, Leaf Disease with Background present – DSRP Retrieval Method.

Figure 6.23 clearly demonstrates that the lightweight model achieves flawless retrieval, as every returned image belongs to the correct class with no misclassifications.

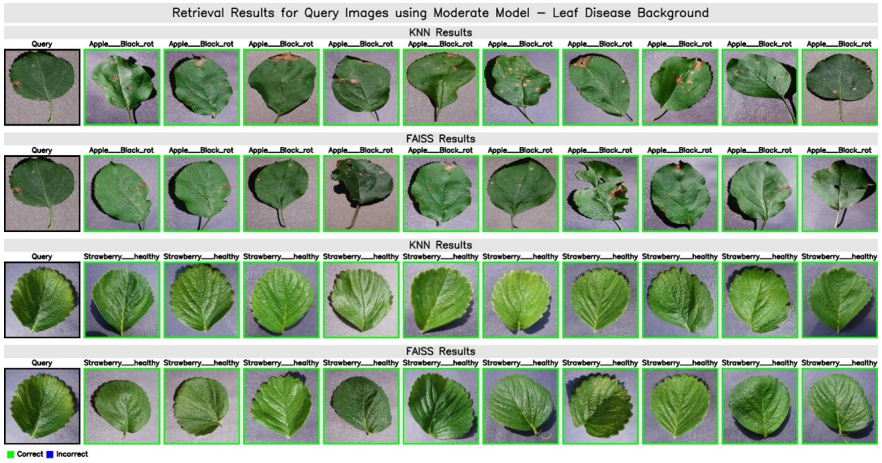

Figure 6.24: Retrieval Results for Query Images using the Moderate Model, Leaf Disease with Backgroundpresent – DSRP Retrieval Method.

In Figure 6.24, the moderate model demonstrates flawless retrieval performance, returning only images from the correct class with no misclassifications observed.

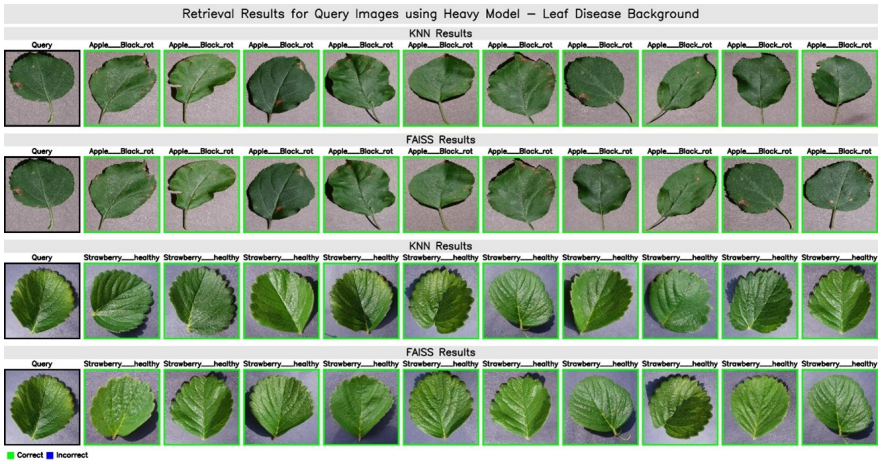

Figure 6.25: Retrieval Results for Query Images using the Heavy Model, Leaf Disease with Background present – DSRP Retrieval Method.

In Figure 6.25, the heavy model exhibits flawless retrieval performance, with every returned image accurately matching the correct class and no misclassifications observed.

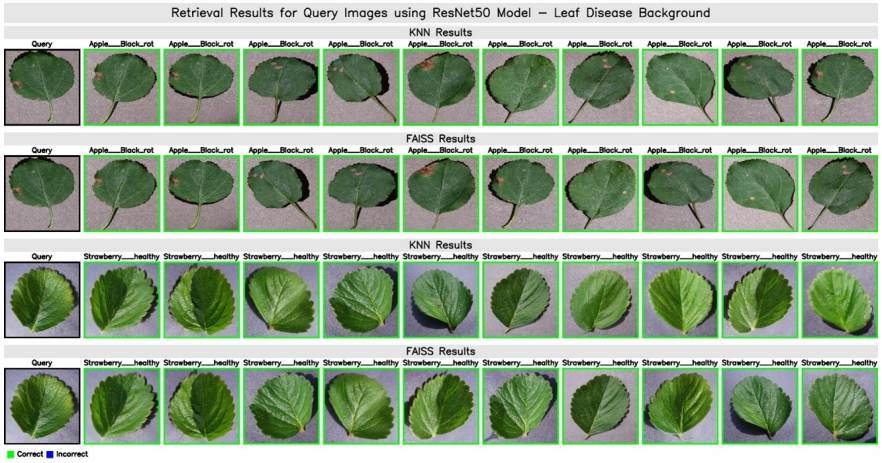

Figure 6.26: Retrieval Results for Query Images using the ResNet50 Model, Leaf Disease with Background present – DSRP Retrieval Method.

In Figure 6.26, the ResNet50 model exhibits flawless retrieval performance, with every returned image matching the correct class and no misclassifications observed.

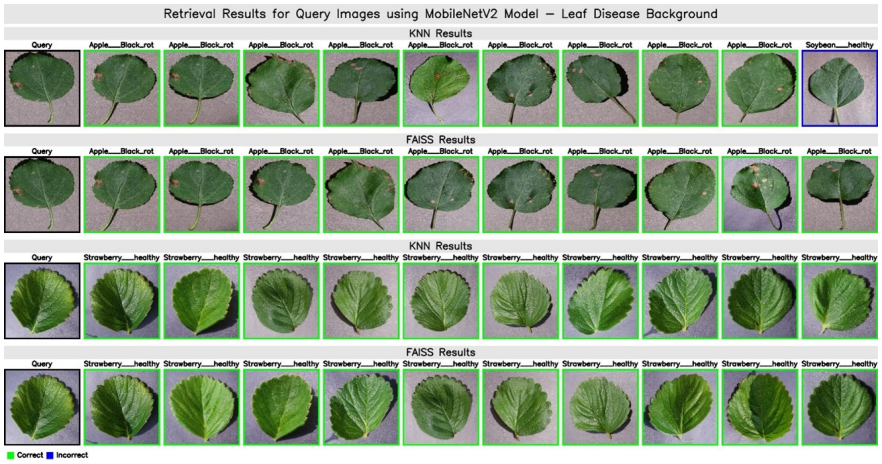

Figure 6.27: Retrieval Results for Query Images using the MobileNetV2 Model, Leaf Disease with Background present – DSRP Retrieval Method.

In Figure 6.27, the MobileNetV2 model demonstrates near perfect retrieval performance. The majority of returned images belong to the correct class, with a single misclassification, Soybean Healthy,appearing at the end of the retrieval list, indicating that it has the greatest distance from the query image.

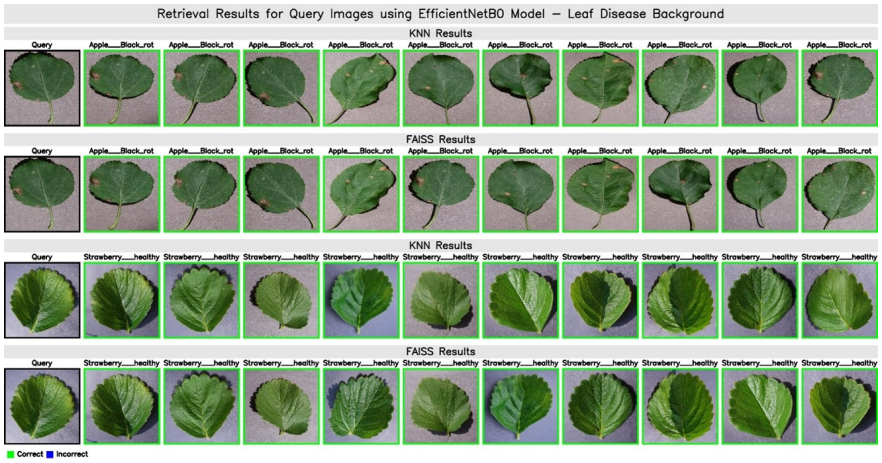

Figure 6.28: Retrieval Results for Query Images using the EfficientNetB0 Model, Leaf Disease with Background present – DSRP Retrieval Method.

In Figure 6.28, the EfficientNetB0 model demonstrates perfect retrieval performance, with every returned image belonging to the correct class and no misclassifications observed.

Leaf Disease (No Background) (6.6.1.2)

The section below shows the retrieval performance for the Leaf Disease dataset with no background present.

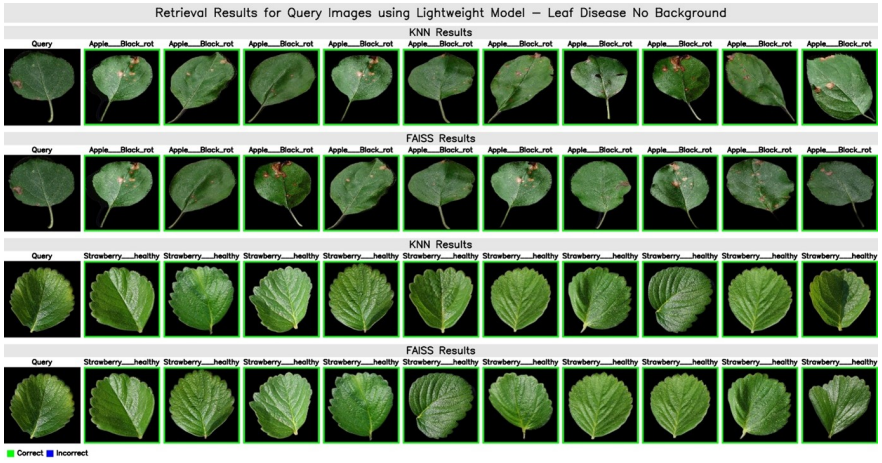

Figure 6.29: Retrieval Results for Query Images using the Lightweight Model, Leaf Disease with No Background present – DSRP Retrieval Method.

In Figure 6.29, the lightweight model achieves flawless retrieval, with all returned images belonging to the correct class and no misclassifications observed.

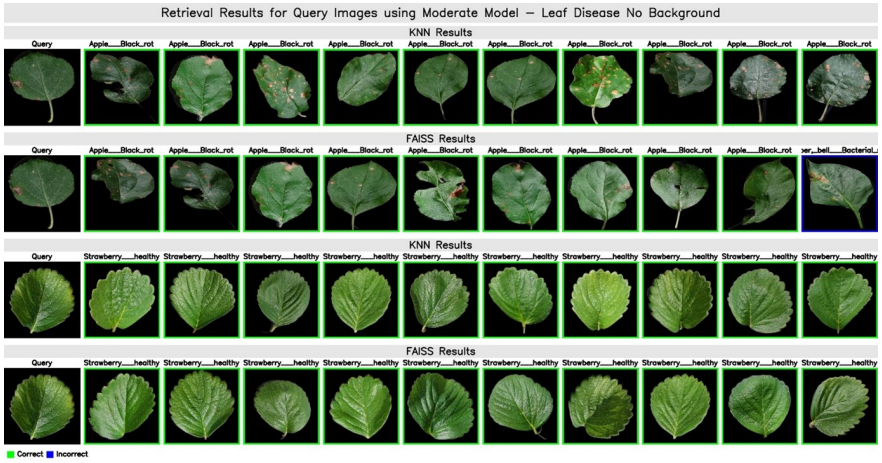

Figure 6.30: Retrieval Results for Query Images using the Moderate Model, Leaf Disease with No Background present – DSRP Retrieval Method.

Above in Figure 6.30 the moderate model achieves perfect retrieval. All the correct classes are returned and no classifications is observed.

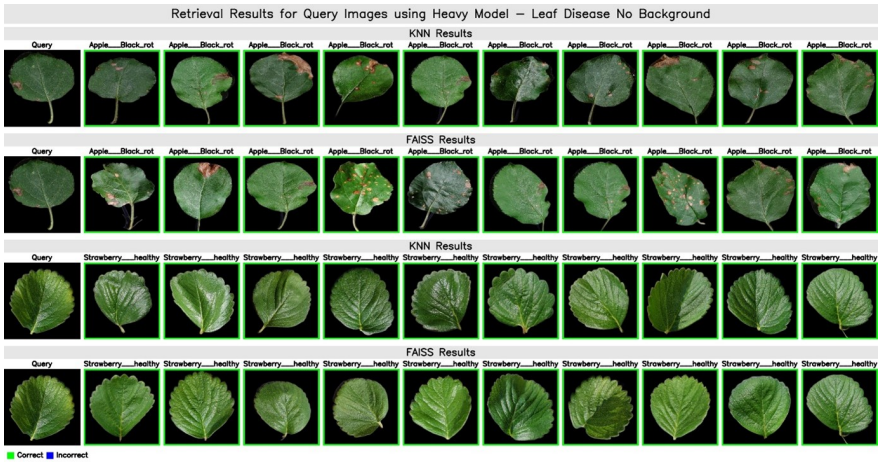

Figure 6.31: Retrieval Results for Query Images using the Heavy Model, Leaf Disease with No Background present – DSRP Retrieval Method.

In Figure 6.31, the heavy model exhibits flawless retrieval performance, with every returned image accurately belonging to the correct class and no misclassifications observed.

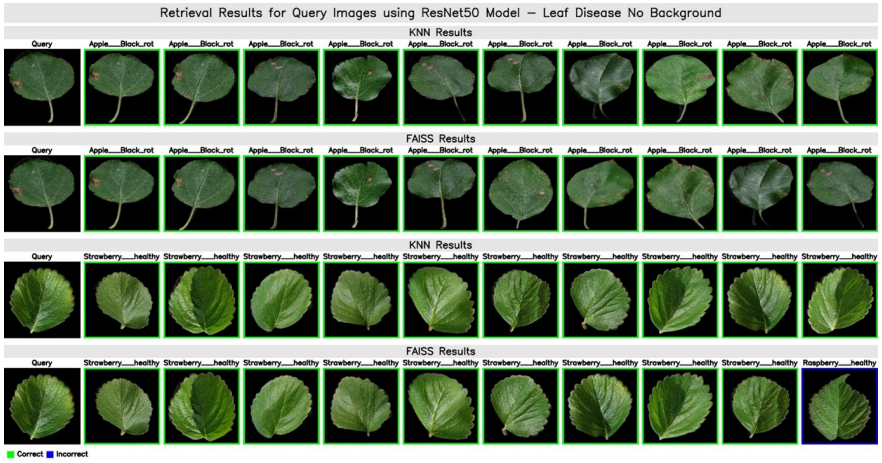

Figure 6.32: Retrieval Results for Query Images using the ResNet50 Model, Leaf Disease with No Background present – DSRP Retrieval Method.

In Figure 6.32, the ResNet50 model achieves perfect retrieval performance, with all returned images belonging to the correct class and no misclassifications observed.

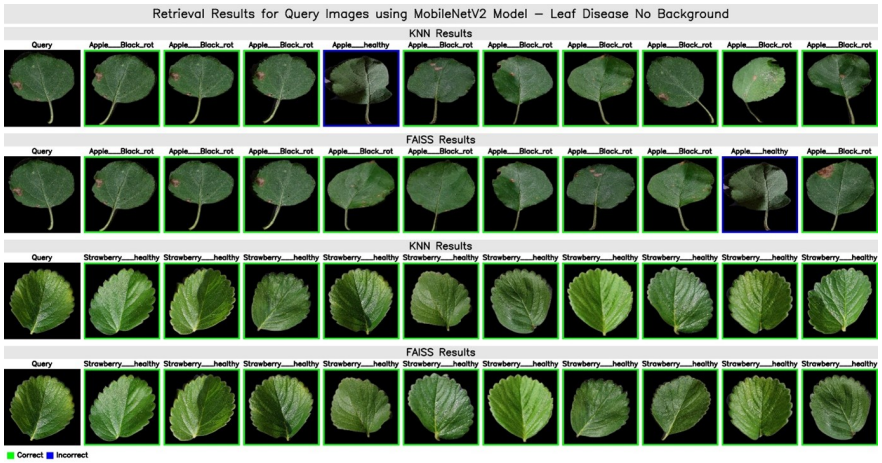

Figure 6.33: Retrieval Results for Query Images using the MobileNetV2 Model, Leaf Disease with No Background present – DSRP Retrieval Method.

In Figure 6.33, the MobileNetV2 model demonstrates perfect retrieval performance, with every returned image belonging to the correct class and no misclassifications observed.

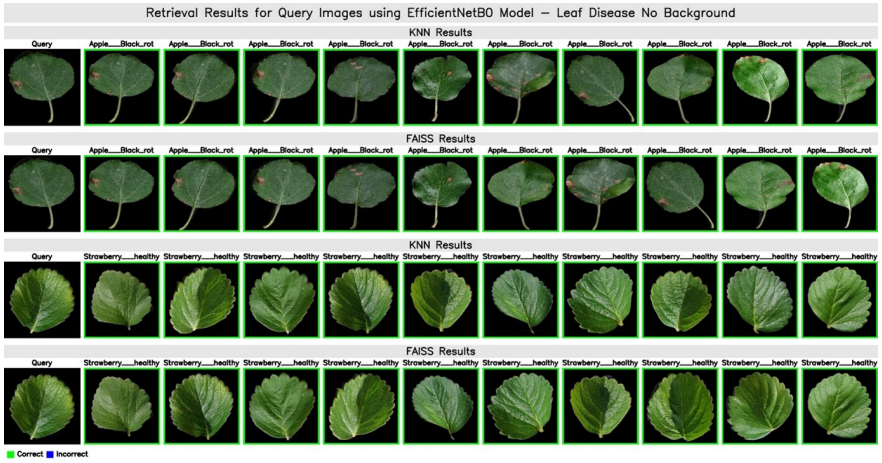

Figure 6.34: Retrieval Results for Query Images using the EfficientNetB0 Model, Leaf Disease with No Background present – DSRP Retrieval Method.

In Figure 6.34, the EfficientNetB0 model demonstrates perfect retrieval performance, with all returned images accurately belonging to the correct class and no misclassifications observed.

Brain Tumour MRI Scans (6.6.2.1)

The section below shows the retrieval performance for the Brain Tumour MRI Scans dataset with the background present.

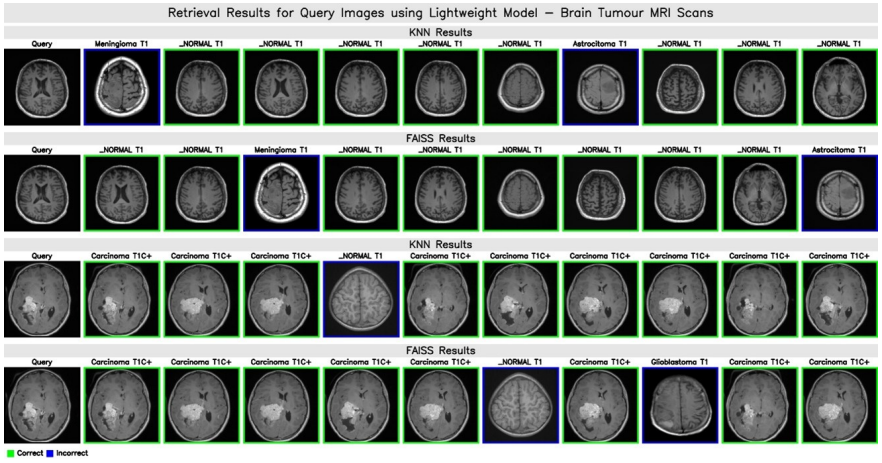

Figure 6.35: Retrieval Results for Query Images using the Lightweight Model, Brain Tumour MRI Scans with Background present – DSRP Retrieval Method.

In Figure 6.35, the lightweight model demonstrates near-perfect retrieval performance. While most of the retrieval results are correct, a few misclassifications, spanning different classes, are observed in both the KNN and FAISS retrieval outputs.

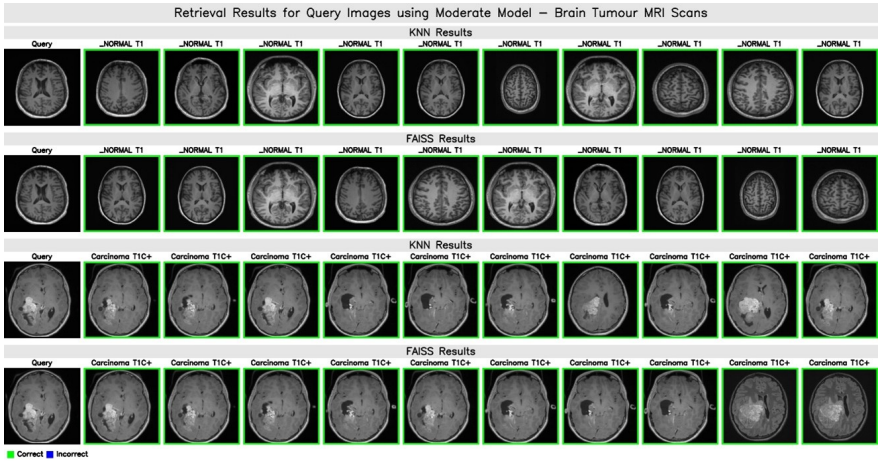

Figure 6.36: Retrieval Results for Query Images using the Moderate Model, Brain Tumour MRI Scans with Background present – DSRP Retrieval Method.

In Figure 6.36, the moderate model exhibits flawless retrieval performance, with every returned image belonging to the correct class and no misclassifications observed.

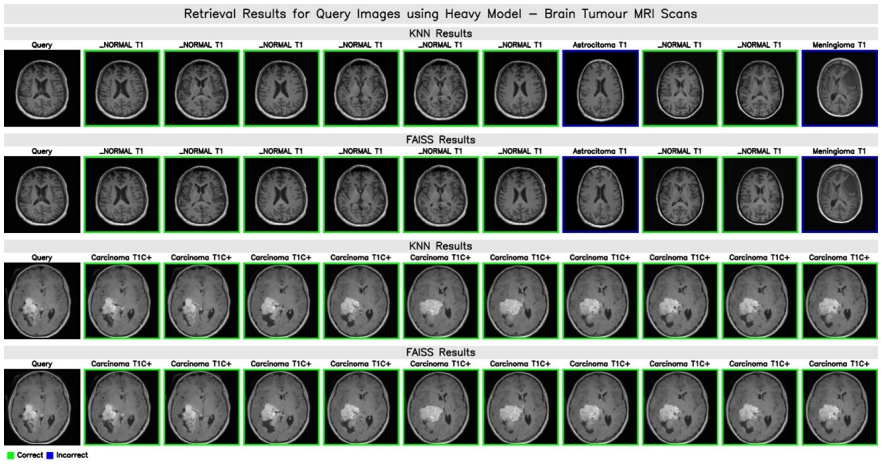

Figure 6.37: Retrieval Results for Query Images using the Heavy Model, Brain Tumour MRI Scans with Background present – DSRP Retrieval Method.

In Figure 6.37, the heavy model demonstrates near perfect retrieval performance. The majority of the results are accurate, though a few misclassifications are observed in the KNN outputs, specifically retrieving the Astrochloma T1 and Meningioma T1 classes.

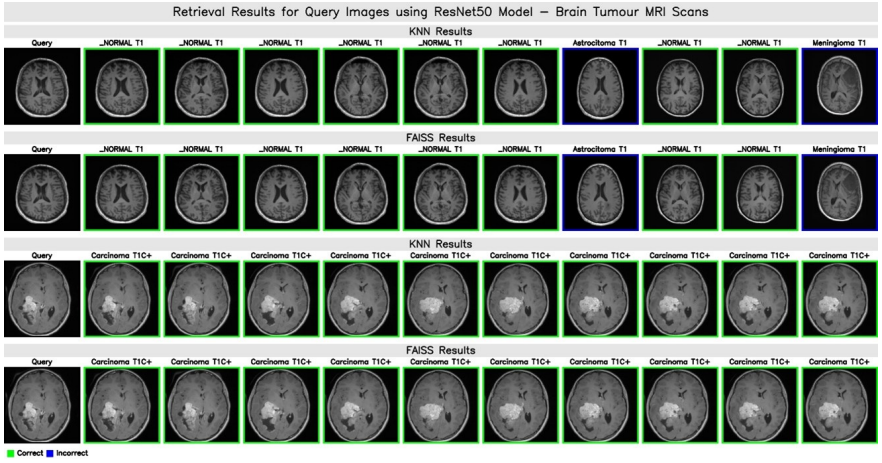

Figure 6.38: Retrieval Results for Query Images using the ResNet50 Model, Brain Tumour MRI Scans with Background present – DSRP Retrieval Method.

In Figure 6.38, the ResNet50 model demonstrates near-perfect retrieval performance. Although most results are accurate, a few misclassifications are observed in the KNN outputs, specifically retrieving the Astrochloma T1 and Meningioma T1 classes, similar to the heavy model, Figure 6.37.

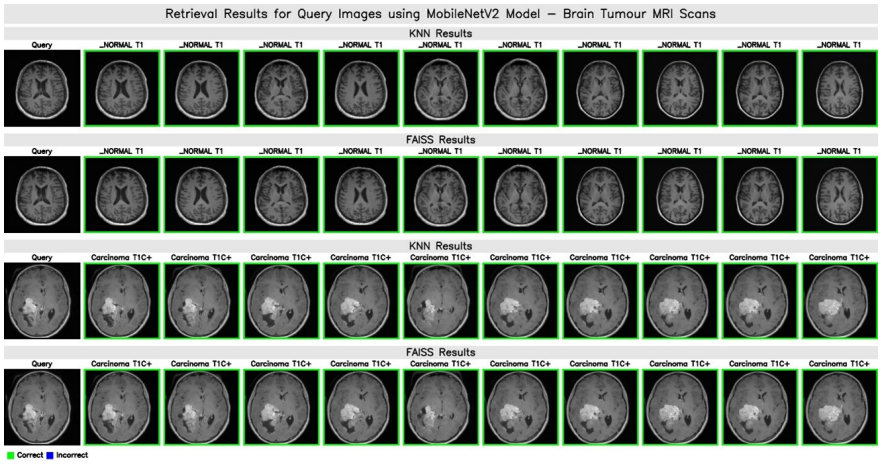

Figure 6.39: Retrieval Results for Query Images using the MobileNetV2 Model, Brain Tumour MRI Scans with Background present – DSRP Retrieval Method.

In Figure 6.39, the MobileNetV2 model demonstrates perfect retrieval performance, with all returned images accurately matching the correct class and no misclassifications observed.

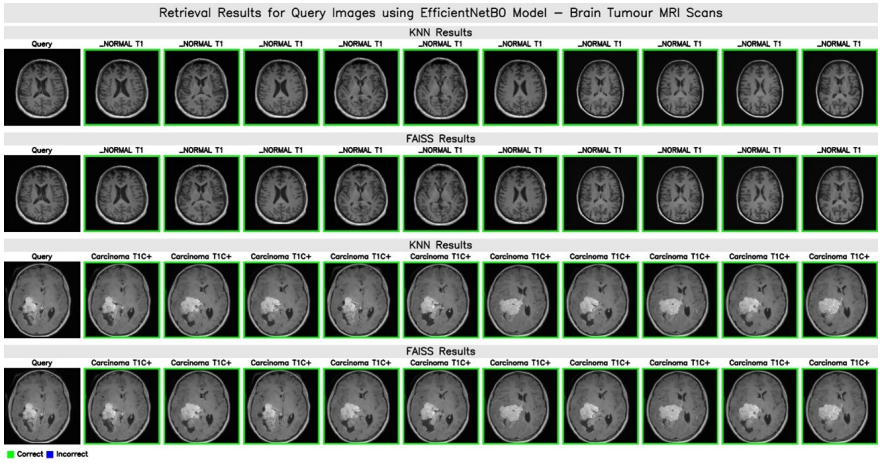

Figure 6.40: Retrieval Results for Query Images using the EfficientNetB0 Model, Brain Tumour MRI Scans with Background present – DSRP Retrieval Method.

In Figure 6.40, the EfficientNetB0 model demonstrates perfect retrieval performance, with all returned images accurately matching the correct class and no misclassifications observed.

COVID-19 Scans (6.6.3.1)

The section below shows the retrieval performance for the COVID-19 Scans dataset with the background present.

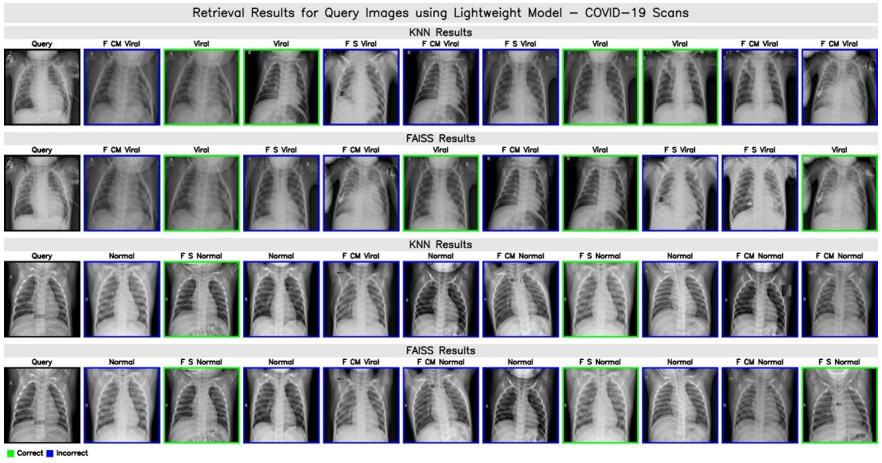

Figure 6.41: Retrieval Results for Query Images using the Lightweight Model, COVID-19 Scans with Background present – DSRP Retrieval Method.

In Figure 6.41, the lightweight model demonstrates substantial misclassification, with numerous incorrect classes among the top retrieval results.

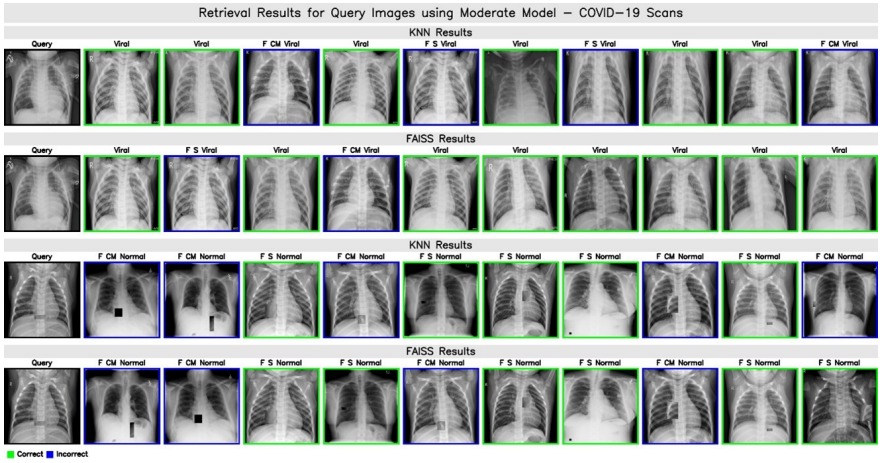

Figure 6.42: Retrieval Results for Query Images using the Moderate Model, COVID-19 Scans with Background present – DSRP Retrieval Method.

In Figure 6.42, the moderate model shows improved retrieval performance compared to the lightweight model, Figure 6.41, with fewer misclassifications observed overall, although a few errors still persist.

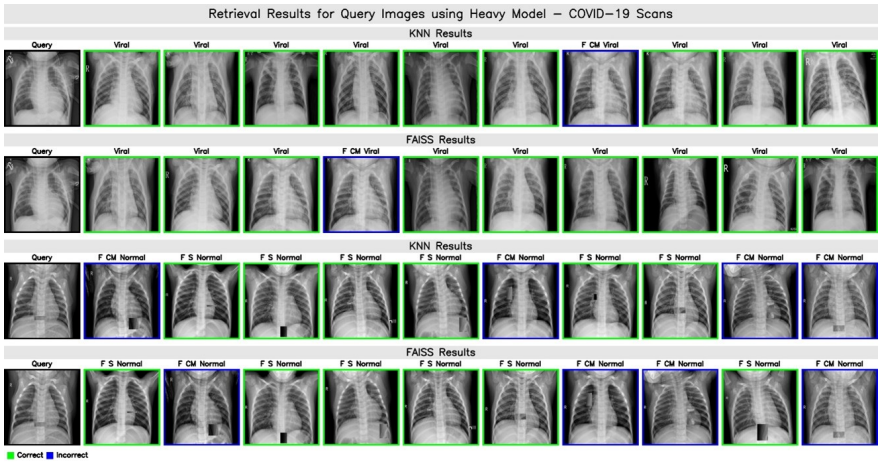

Figure 6.43: Retrieval Results for Query Images using the Heavy Model, COVID-19 Scans with Background present – DSRP Retrieval Method.

In Figure 6.43, the heavy model shows strong retrieval performance overall. However, misclassifications are observed, with the FAISS method retrieving F CM Normal and the KNN approach retrieving F CM Viral incorrectly.

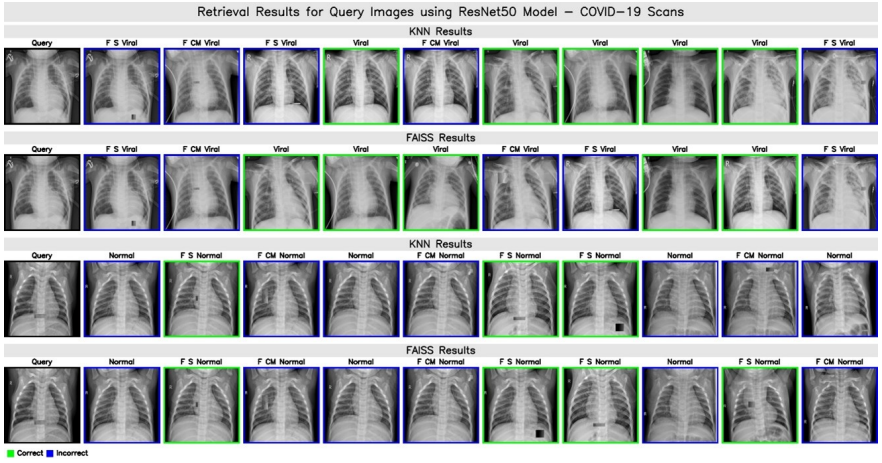

Figure 6.44: Retrieval Results for Query Images using the ResNet50 Model, COVID-19 Scans with Background present – DSRP Retrieval Method.

In Figure 6.44, the ResNet50 model exhibits a higher rate of misclassifications compared to the moderate, Figure 6.42, and heavy models, Figure 6.43, With errors spanning a diverse range of classes.

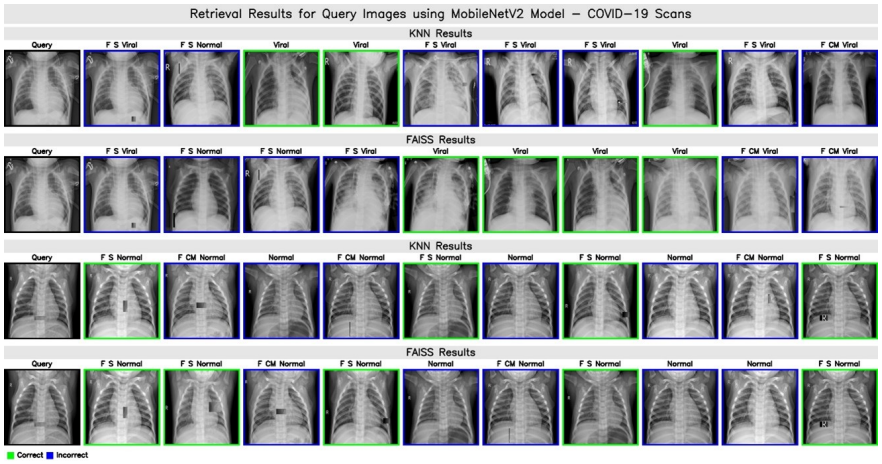

Figure 6.45: Retrieval Results for Query Images using the MobileNetV2 Model, COVID-19 Scans with Background present – DSRP Retrieval Method.

In Figure 6.45, the MobileNetV2 model shows significant misclassification, with errors spanning a diverse range of classes.

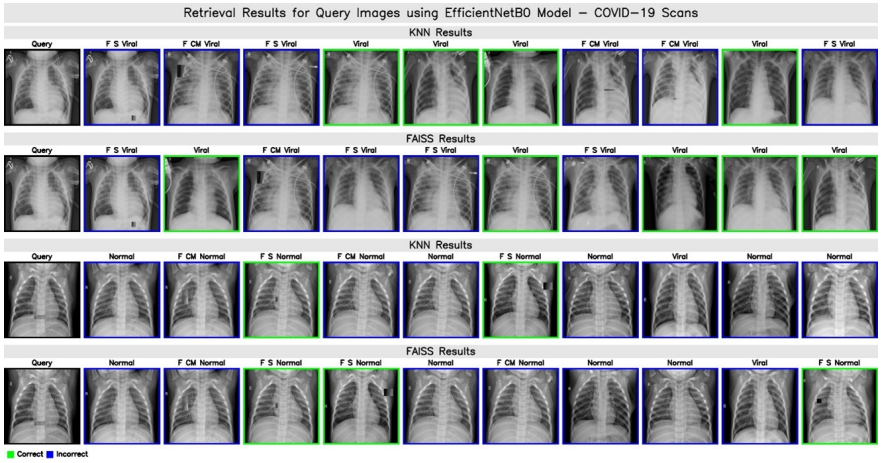

Figure 6.46: Retrieval Results for Query Images using the EfficientNetB0 Model, COVID-19 Scans with Background present – DSRP Retrieval Method.

In Figure 6.46, the EfficientNetB0 model exhibits significant misclassification, similar to the ResNet50, Figure 6.44, and MobileNetV2, Figure 6.45 models. With errors spanning a diverse range of classes.

Pokémon (6.6.4.1)

The section below shows the retrieval performance for the Pokemon dataset with the background present.

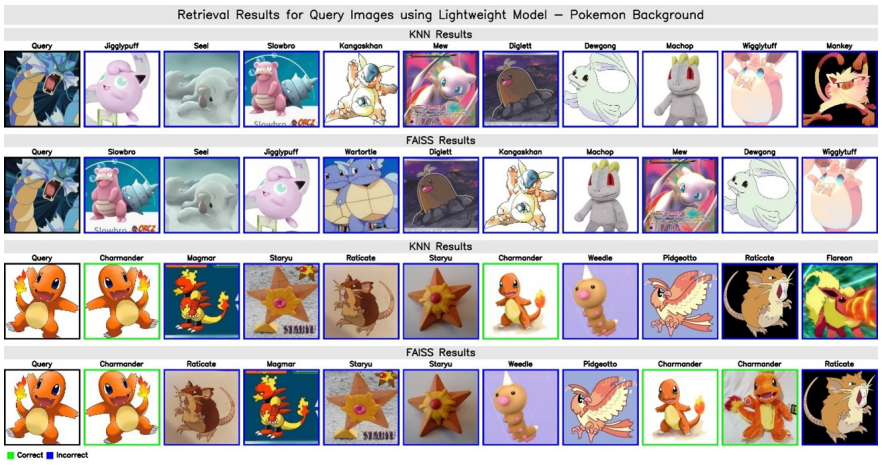

Figure 6.47: Retrieval Results for Query Images using the Lightweight Model, Pokémon with Background present – DSRP Retrieval Method.

In Figure 6.47, the lightweight model displays varied performance across the two query images. For the first query image, all retrievals are incorrect, whereas for the second query image, a few correct classifications are achieved.

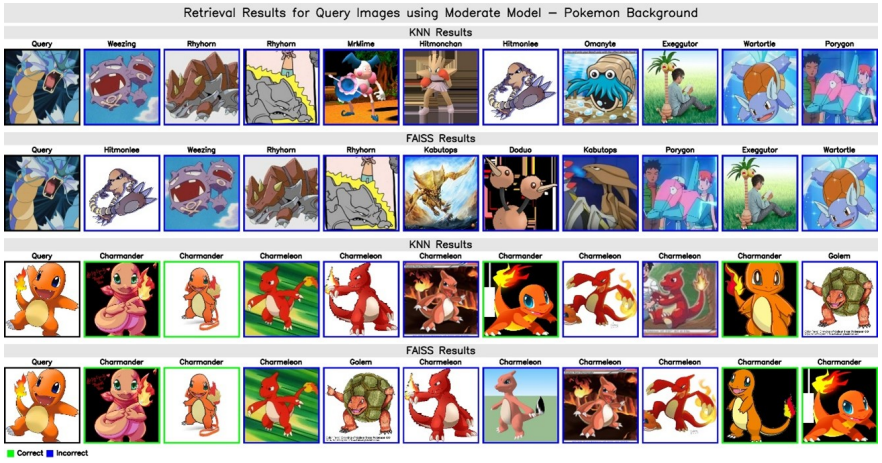

Figure 6.48: Retrieval Results for Query Images using the Moderate Model, Pokémon with Background present – DSRP Retrieval Method.

In Figure 6.48, the moderate model continues to show a pattern where the first query image yields entirely incorrect classifications. For the second query image, while a few correct classifications are observed, there is also a noticeable prevalence of visual patterns from the incorrect classifications.

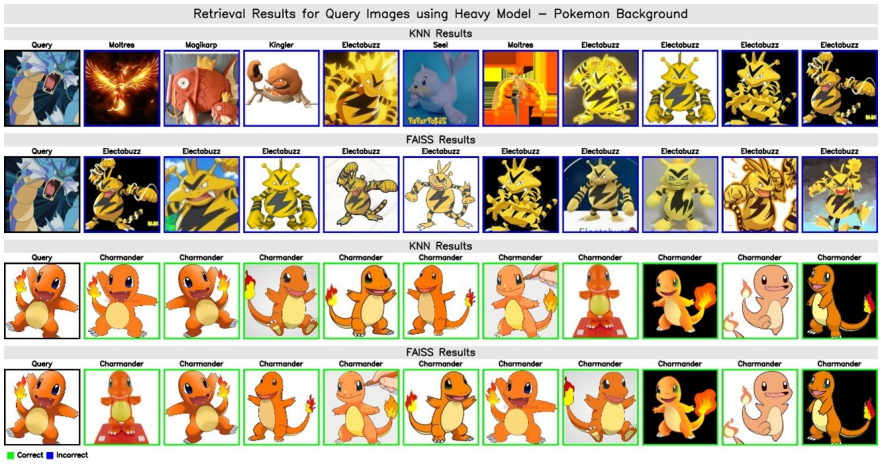

Figure 6.49: Retrieval Results for Query Images using the Heavy Model, Pokémon with Background present – DSRP Retrieval Method.

In Figure 6.49, the heavy model still returns entirely incorrect classifications for the first query image, whereas it achieves perfect retrieval for the second query image, with all results correctly classified.

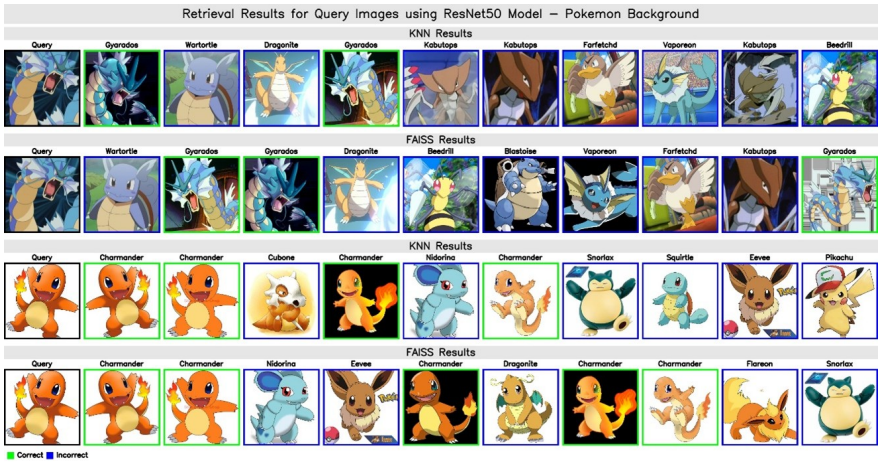

Figure 6.50: Retrieval Results for Query Images using the ResNet50 Model, Pokémon with Background present – DSRP Retrieval Method.

In Figure 6.50, the ResNet50 model shows a mixed performance. For the first query image, a few correct classifications are observed for the first time, while for the second query image, both correct and incorrect results are evident.

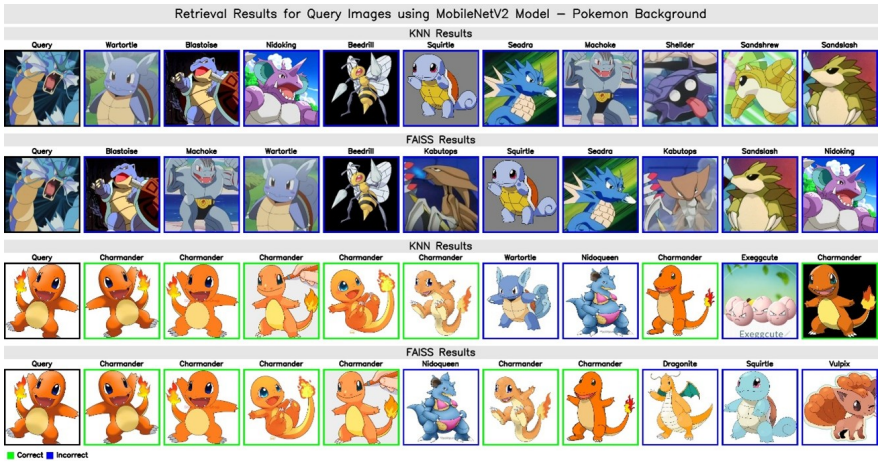

Figure 6.51: Retrieval Results for Query Images using the MobileNetV2 Model, Pokémon with Background present – DSRP Retrieval Method.

In Figure 6.51, the MobileNetV2 model retrieves entirely incorrect classifications for the first query image, while the second query image shows an improvement with more correct classifications, although some errors still persist.

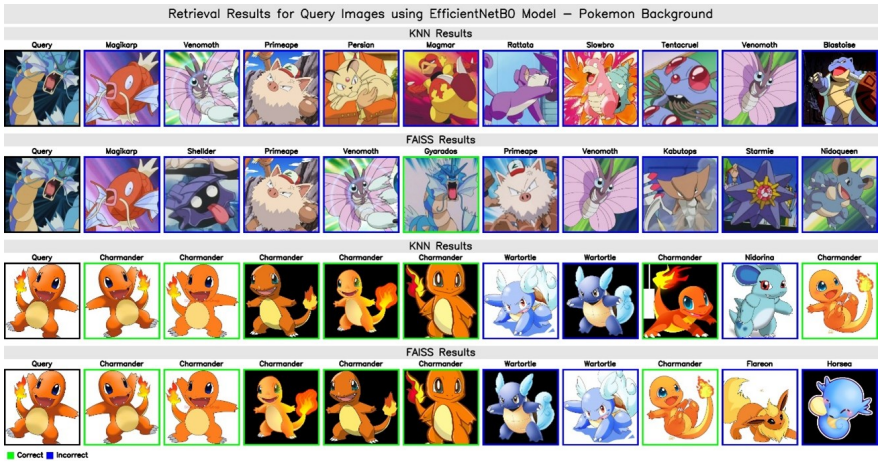

Figure 6.52: Retrieval Results for Query Images using the EfficientNetB0 Model, Pokémon with Background present – DSRP Retrieval Method.

In Figure 6.52, the EfficientNetB0 model retrieves entirely incorrect classifications for the first query image, whereas the second query image shows an improvement with more correct classifications, though not all results are accurate.

Pokémon (No Background) (6.6.4.2)

In this section, we analyze the impact of background removal on retrieval performance for the Pokemon dataset. Notably, the first query image, processed with background removal, illustrates a case where the Rembg tool inadvertently removed some of the image's most distinctive features.

This degradation in visual cues makes the image significantly harder to classify, highlighting that background removal techniques do not always produce optimal results.

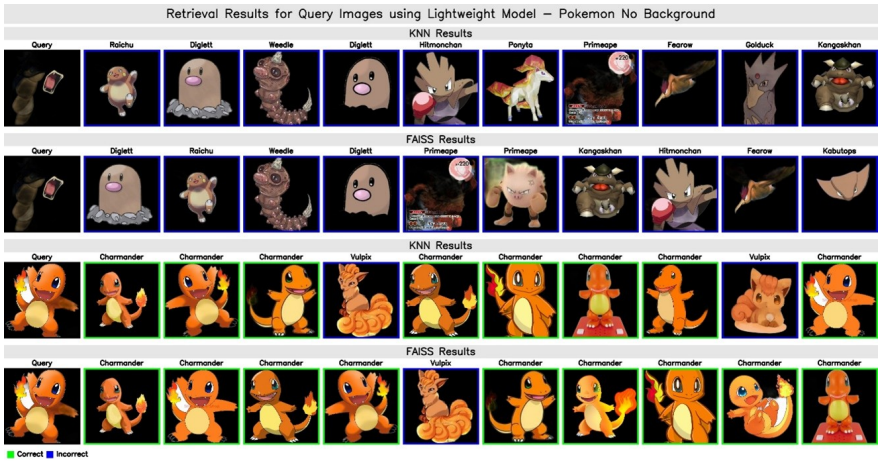

Figure 6.53: Retrieval Results for Query Images using the Lightweight Model, Pokémon with No Background present – DSRP Retrieval Method.

In Figure 6.53, the lightweight model, without background, retrieves entirely incorrect classifications for the first query image, while the second query image yields only correct classifications.

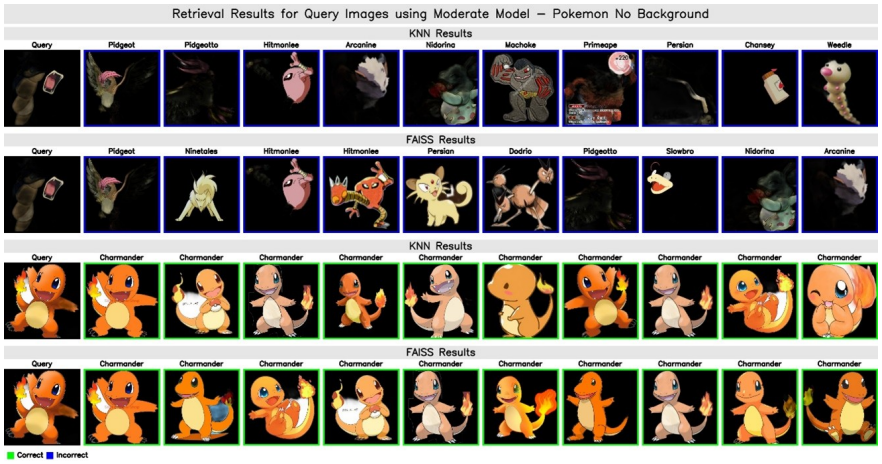

Figure 6.54: Retrieval Results for Query Images using the Moderate Model, Pokémon with No Background present – DSRP Retrieval Method.

In Figure 6.54, the moderate model , without background, exhibits the same trend, all classifications for the first query image are incorrect, while all classifications for the second query image are correct.

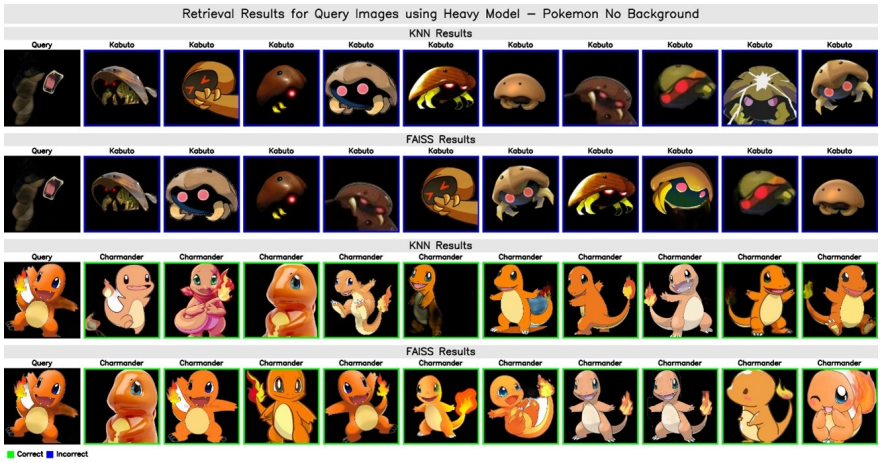

Figure 6.55: Retrieval Results for Query Images using the Heavy Model, Pokémon with No Background present – DSRP Retrieval Method.

In Figure 6.55, the heavy model, without background, shows that all retrievals for the first query are incorrect, even though the retrieved images bear a close visual resemblance to the input. Conversely, all classifications for the second query are correct.

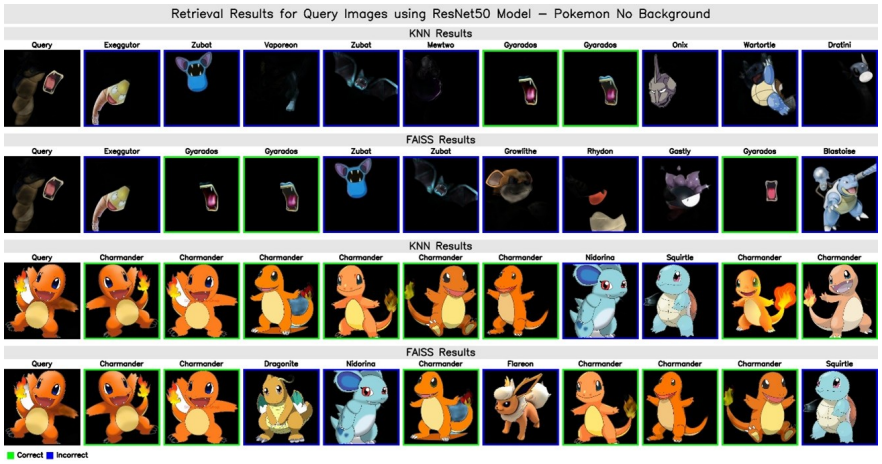

Figure 6.56: Retrieval Results for Query Images using the ResNet50 Model, Pokémon with No Background present – DSRP Retrieval Method.

In Figure 6.56, some correct classifications are observed for the first time, and some of the retrieved images exhibit somewhat similar features to the query image. For the second query, the majority of classifications are correct, but some misclassifications occur, with the incorrect results displaying some resemblance to the query.

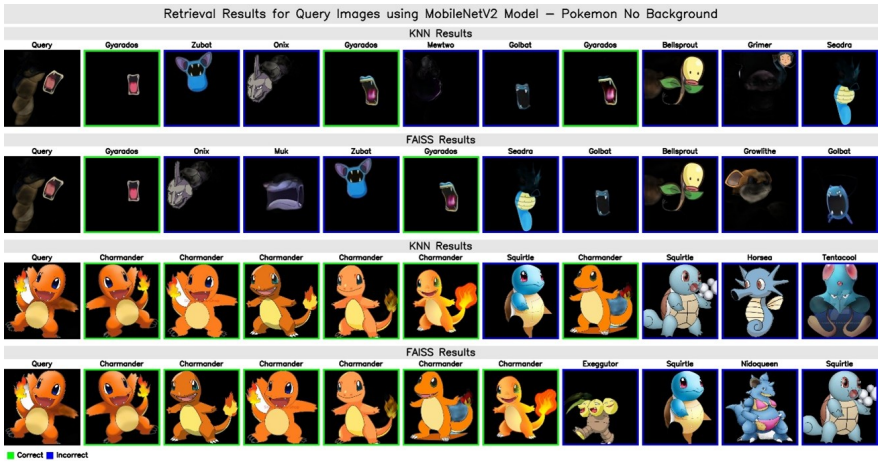

Figure 6.57: Retrieval Results for Query Images using the MobileNetV2 Model, Pokémon with No Background present – DSRP Retrieval Method.

In Figure 6.57, for the first query image, some correct classifications are observed while other retrieved images exhibit similar features to the query. For the second query image, the majority of the retrievals are correct, although the misclassified results lack colour similarity in features to the query but similiar shape.

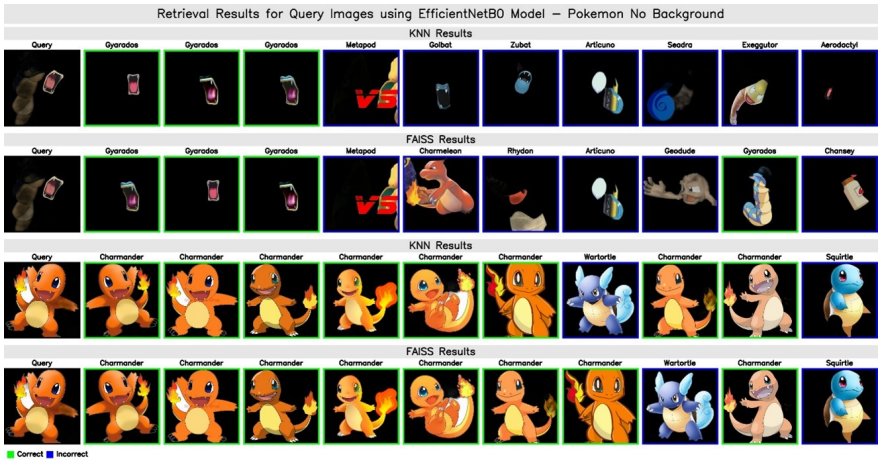

Figure 6.58: Retrieval Results for Query Images using the EfficientNetB0 Model, Pokémon with No Background present – DSRP Retrieval Method.

In Figure 6.58, for the first query image, some correct classifications are observed while other retrieved images exhibit features similar to the query. For the second query image, the majority of the retrievals are correct, although the misclassified results share similar shape features with the query.

6.7.2 Retrieval Results Post Analysis

Overall, the results demonstrate that the retrieval pipelines perform exceptionally well on the Leaf Disease dataset. With the background present, all evaluated models, including the lightweight, moderate, heavy, ResNet50, and EfficientNetB0 architectures, achieve flawless retrieval, while MobileNetV2 attains near perfect performance with only a minor misclassification at the far end of the retrieval list. When background removal is applied, as illustrated in Figures 6.29 through 6.34, every model retrieves only the correct classes, underscoring that the key distinguishing features in the Leaf Disease images remain well preserved despite the removal of background information.

In contrast, the Brain Tumour MRI dataset yields near perfect retrieval overall. As seen in Figures 6.35 through 6.40, the moderate, MobileNetV2, and EfficientNetB0 models achieve perfect classification, while the lightweight model shows minor misclassifications across both the KNN and FAISS pipelines. The heavy and ResNet50 models perform strongly, though occasional misclassifications, particularly involving the Astrochloma T1 and Meningioma T1 classes, suggest that subtle variations in model complexity can influence retrieval accuracy.

For COVID-19 scans with background, the performance is more varied. The heavy model, Figure 6.43 demonstrates strong retrieval performance despite misclassifications (e.g., FAISS misclassifying F CM Normal and KNN misclassifying F CM Viral), while the ResNet50 model, Figure 6.44 exhibits a higher rate of errors across diverse classes. Both the MobileNetV2 and EfficientNetB0 models (Figures 6.45 and 6.46, respectively) show significant misclassifications, indicating that the COVID-19 dataset presents inherent challenges for accurate retrieval under the domain specific pipeline.

The Pokémon dataset reveals the most varied outcomes. With background present, as depicted in Figures 6.47 through 6.52, the lightweight and moderate models fail entirely on the first query image, although the moderate model shows some correct classifications for the second query despite recurring error patterns. The heavy model returns completely incorrect results for the first query but achieves perfect accuracy for the second, while the ResNet50 model displays mixed performance. In the absence of background highlighted in Figures 6.53 through 6.58, the first query image is typically misclassified, likely due to the removal of distinctive features by the Rembg tool. Conversely, the second query is generally well retrieved across most models, though occasional errors persist.

In summary, while models exhibit excellent performance on the Leaf Disease dataset regardless of background presence, the challenges in the COVID-19 and Pokémon datasets become more pronounced, particularly when background removal obscures critical image features. The Brain Tumour MRI dataset further illustrates that even minor differences in model architecture can lead to subtle variations in classification accuracy. For detailed visual comparisons, refer to the key figures

provided for each dataset and experimental condition.

Chapter 7

Discussion

7.0.1 Introduction

Embedding-based image retrieval systems are also promising solutions in most applications, particularly in healthcare and agriculture, where accurate retrieval results, computing efficiency, and flexibility are of the most concern. These systems are built upon deep learning models and high-dimensional feature embeddings and enable accurate and scalable similar image retrieval and are utilized for assisting medical diagnosis, crop monitoring, and machine decision-making [172, 173].

One of the advantages of such retrieval pipelines is that they can generalize computationally effectively across high-dimensional datasets. However, bias reduction, retrieval explainability, and adversarial robustness remain open problems, particularly in high-stakes applications such as clinical decision support and precision agriculture [174]. These are problems that must be resolved by balancing accuracy, computational tractability, and ethical AI principles.

This work contrasts two disparate retrieval pipelines, the Domain-Specific Retrieval Pipeline (DSRP) and the Pre-trained Feature Extraction Pipeline (PMRP), across four various datasets with diverse retrieval difficulties. Contrast between approaches illustrates how embedding-based systems are affected by disparate real-world conditions, from insidious medical image variation to regimented benchmark circumstances.

7.0.2 Importance of Dataset Diversity and Selection

The choice of dataset is a key to the performance and generalization of embedding-based image retrieval systems. A heterogeneous dataset ensures strong and transferable features learned by models, improving the accuracy of retrieval on various domains [175]. Dataset validity and variance are particularly crucial in retrieval applications that have large intra-class variations and subtle inter-class differences.

Dataset Diversity and Its Impact

- **Generalization Across Domains**: A dataset with sufficient diversity allows models to

generalize beyond specific contexts, reducing bias and improving real-world applicability [91].

- **Feature Learning and Representation**: The inclusion of varied image structures ensures that deep learning models capture meaningful and transferable embeddings [176].
- **Bias Mitigation**: Limited dataset diversity can introduce unintended biases, leading to suboptimal performance in applications such as medical image retrieval or precision agriculture [177].

For this research, dataset selection was carefully designed to ensure a comprehensive assessment of retrieval performance. The chosen datasets span medical imaging, agriculture, and general classification tasks, each presenting unique challenges:

- **Medical Image Retrieval**: Requires fine-grained feature extraction due to subtle pathological variations.
- **Agricultural Image Analysis**: Demands robust domain adaptation due to environmental factors.
- **General Object Retrieval**: Serves as a benchmark for retrieval efficiency and model scalability.

In addition to qualitative selection, dataset imbalance ratios and class distributions were analyzed so that retrieval accuracy will not be biased towards majority classes. Prior work has emphasized the need for structured dataset variance analysis in retrieval-based AI applications [93].

By incorporating a diverse dataset pool, this study ensures that retrieval models are tested across multiple real-world challenges, enhancing their robustness and applicability.

7.0.3 Findings

The technical evaluation of the two pipelines reveals several quantitative insights that are critical for designing efficient image retrieval systems. In the Domain-Specific Retrieval Pipeline (DSRP), domain-specific training and model fine-tuning are key to optimizing retrieval performance across datasets. For example, in the Brain Tumour MRI dataset, moderate models achieve a top-10 accuracy of 0.92 with precision at 5 (P@5) of 0.80 and F1 scores around 0.83, while lightweight models yield a slightly lower top-10 accuracy of 0.85 and F1 scores near 0.55. Similar trends are observed in the COVID-19 Scans dataset, where the moderate variant also attains a top-10 accuracy of 0.93 with P@5 values of 0.62, in contrast to the lightweight variant's P@5 around 0.38. These metrics

underscore the impact of model complexity on retrieval performance, with moderate and heavy configurations delivering consistently superior precision, recall, and F1 scores compared to their lightweight counterparts. In terms of inference efficiency, the DSRP demonstrates that using FAISS significantly reduces retrieval latency; for instance, lightweight models show an inference time improvement from 0.1006 seconds with KNN to 0.0881 seconds with FAISS.

In contrast, the Pre-trained Feature Extraction Pipeline (PMRP) leverages robust models such as ResNet50, MobileNetV2, and EfficientNetB0 to produce highly discriminative embeddings without the need for extensive domain-specific adjustments. This approach yields near-perfect top-10 accuracies in the Leaf Disease dataset, with accuracies reaching approximately 0.99 and F1 scores above 0.95. In the Brain Tumour MRI and COVID-19 Scans datasets, PMRP maintains high performance with top-10 accuracies typically between 0.95 and 0.97 and consistent precision and recall values that parallel those observed in the DSRP, though with less variability. The uniformity in these metrics indicates that pre-trained models, by virtue of their exposure to large-scale datasets, can generalize effectively across different domains. Moreover, PMRP benefits from a consistent inference time profile; for example, the inference times for ResNet50 in the PMRP are recorded at approximately 0.1180 seconds with KNN and 0.1197 seconds with FAISS, showcasing the pipeline's stability across diverse applications.

These numerical findings highlight the trade-offs between customization and consistency. The DSRP's tailored approach allows for fine-tuning that optimizes performance for specific datasets, as evidenced by the marked differences in metrics between lightweight and heavier models. However, this customization introduces variability and increased computational overhead. On the other hand, the PMRP provides a robust baseline that achieves uniformly high accuracy and efficiency across multiple datasets, making it a more scalable solution for real-time applications. The consistent reduction in inference time using FAISS across both pipelines further emphasizes the importance of selecting efficient retrieval algorithms, particularly when deploying systems in high-stakes environments such as medical diagnostics.

Overall, these findings offer a comprehensive technical perspective, demonstrating that while domain-specific tuning in the DSRP can lead to significant improvements in retrieval metrics under certain conditions, the PMRP's reliance on pre-trained models provides a consistently high-performance alternative. This nuanced understanding of performance trade-offs, backed by quantitative metrics, lays a solid foundation for future research and practical applications in trustworthy AI systems.

7.0.4 Considerations and Ethics

Deployment of embedding-based image retrieval systems to high-stakes environments not only requires high performance and computational efficiency but also poses ethical and practical issues that need to be considered with meticulous thoughtfulness. Algorithmic bias is a primary concern, which could stem from data imbalance. In our Domain-Specific Retrieval Pipeline (DSRP), the utilization of high-quality labeled datasets as the basis guarantees that even selection bias in the data could result in biased retrieval outputs that have an unequal representation of majority classes. In order to address this issue, upcoming research has to incorporate fairness-aware approaches, such as domain adaptation and adversarial debiasing, to ensure fair performance for every class.

The second fundamental challenge is the inherent opaqueness of deep feature embeddings, which makes it difficult for end-users to understand why a particular retrieval result is obtained. Greater transparency and interpretability are essential, particularly in sensitive domains like healthcare and legal use cases.

Embedding explainable AI (XAI) techniques, such as attention visualizations (e.g., Grad-CAM, LRP) and saliency maps, can unveil significant information about the decision-making process [178, 179]. These methods provide a way to trace back how specific features contribute to similarity scores, enhancing user trust in the system's outputs [180, 181]. Additionally, prototype-based retrieval models and feature attribution techniques like SHAP and LIME help interpret latent space representations in deep embedding-based retrieval systems [182, 183].

Aside from transparency and fairness, the robustness and security of these retrieval models against adversarial attacks is also of paramount importance. For medical and forensic use cases, adversarial manipulations can lead to false image matches, with catastrophic consequences. Therefore, future research must ensure that it gives due attention to enhancing adversarial robustness through methods such as adversarial training and robust feature extraction techniques so that the system is still reliable even in hostile environments.

This paper demonstrates that embedding-based image retrieval systems offer an efficient and scalable solution to image search in healthcare and agricultural applications. The evaluation demonstrates that the Domain-Specific Retrieval Pipeline (DSRP) consistently delivers higher precision for domain-specific applications, while the Pre-trained Feature Extraction Pipeline (PMRP) is a general-purpose and lightweight solution. Despite these successes, bias mitigation, interpretability, and security remain important challenges. Addressing these challenges is essential to developing retrieval systems that are not only accurate but also transparent, fair, and resilient to adversarial attacks.

Chapter 8

Conclusion

This study has provided a comprehensive evaluation of embedding-based image retrieval systems, particularly focusing on their applications in healthcare and agriculture. By comparing the Domain-Specific Retrieval Pipeline (DSRP) and the Pre-trained Feature Extraction Pipeline (PMRP) across multiple datasets, we have demonstrated the nuanced trade-offs between domain-specific fine-tuning and generalization. The findings illustrate that while DSRP excels in specialized applications requiring precise feature extraction, PMRP offers a scalable and efficient alternative that maintains consistently high retrieval performance across diverse datasets.

The technical analysis highlights key performance trade-offs, including the impact of model complexity, dataset characteristics, and retrieval indexing techniques. DSRP benefits from domain-specific adaptation, achieving superior precision in fine-grained medical and agricultural retrieval tasks. However, this comes at the cost of increased computational overhead and variability in performance. Conversely, PMRP leverages pre-trained models to achieve strong and consistent results across datasets, providing a robust and computationally efficient approach suitable for real-time applications [183]. The integration of FAISS indexing further enhances retrieval efficiency, demonstrating the importance of algorithmic optimization in large-scale retrieval systems.

Beyond retrieval accuracy and computational efficiency, this research underscores the critical ethical considerations surrounding bias mitigation, explainability, and adversarial robustness. Algorithmic bias remains a significant challenge, particularly in high-stakes applications where class imbalances can lead to unfair retrieval outputs. Future advancements must prioritize fairness-aware learning techniques, such as domain adaptation and adversarial debiasing, to ensure equitable model performance [182]. Furthermore, the opaqueness of deep feature embeddings raises concerns about interpretability, necessitating the integration of explainable AI (XAI) methodologies, such as attention visualizations and saliency maps, to enhance transparency and user trust [179, 178, 181]. Finally, adversarial robustness remains a pressing concern, particularly in medical and forensic applications, where retrieval errors can have severe consequences. The development of adversarial training strategies and resilient feature extraction techniques is essential to ensuring the security and reliability of these systems.

Overall, this study reinforces the importance of tailoring retrieval strategies to specific domain

requirements while balancing computational feasibility, accuracy, and ethical considerations. By integrating these insights, future work can contribute to the development of more equitable, interpretable, and resilient AI-driven image retrieval systems, with far-reaching implications in healthcare, agriculture, and beyond.

Chapter 9

Future Work and Research

While this study has demonstrated the effectiveness of embedding-based image retrieval systems in healthcare and agriculture, several areas remain open for further exploration. Future research should focus on addressing key challenges related to dataset diversity, bias mitigation, explainability, adversarial robustness, and scalability to enhance the practical applicability and trustworthiness of these retrieval systems.

One of the fundamental areas for future improvement is the expansion of dataset diversity to better represent real-world variations. Ensuring balanced intra-class and inter-class diversity will be essential to prevent bias and improve retrieval performance across different domains [182]. Future studies should explore collecting and curating larger and more representative datasets, developing data augmentation techniques that simulate real-world variations, and investigating synthetic data generation methods such as generative adversarial networks to improve retrieval robustness in underrepresented classes.

Algorithmic bias remains a persistent challenge in image retrieval, particularly in high-stakes domains like medical imaging, where misclassification can have significant consequences [179]. Future research should aim to integrate fairness-aware learning algorithms that address class imbalances and reduce bias in retrieval outcomes. Additionally, developing domain adaptation techniques to ensure equitable retrieval performance across different population groups and exploring adversarial debiasing strategies will be crucial in mitigating unintended biases introduced during model training.

The opacity of deep learning-based retrieval models presents a barrier to their adoption in critical applications. Enhancing model explainability is essential to ensure user trust and regulatory compliance. Future work should focus on implementing explainable AI techniques such as attention visualizations and saliency maps to interpret retrieval decisions [178, 181], developing post-hoc interpretability methods to provide insights into the similarity computations performed by embedding-based models, and creating interactive visualization tools that allow end-users to explore how different features contribute to retrieval outcomes.

As retrieval systems are increasingly deployed in real-world applications, their vulnerability to adversarial attacks must be addressed. Future research should investigate developing adversarial

training techniques to enhance model robustness against manipulative attacks, exploring secure feature extraction methods that ensure the integrity of embeddings in high-stakes applications, and implementing anomaly detection frameworks to identify adversarially altered inputs and mitigate their impact on retrieval results [183].

For embedding-based retrieval systems to be adopted at scale, they must operate efficiently under real-world constraints, particularly in resource-limited environments. Future work should focus on optimizing indexing techniques such as FAISS and approximate nearest neighbor methods to accelerate search efficiency while minimizing memory overhead [183]. Investigating lightweight model architectures, such as MobileNet and EfficientNet variants, for deployment on edge devices and embedded systems will also be crucial. Additionally, exploring distributed retrieval frameworks that leverage cloud and federated learning paradigms can enable large-scale and privacy-preserving retrieval.

Extending embedding-based retrieval beyond single-domain applications can unlock new possibilities in medical diagnostics, agriculture, and other interdisciplinary fields. Future directions include developing multimodal retrieval systems that integrate image embeddings with textual and structured data for richer search capabilities, exploring cross-domain retrieval pipelines that generalize across different imaging modalities, and investigating self-supervised learning approaches to improve retrieval performance without extensive labeled data [182].

Finally, as AI-driven image retrieval systems continue to evolve, ensuring their ethical deployment remains a priority. Future research should focus on developing AI governance frameworks that align retrieval models with ethical principles and regulatory guidelines, investigating human-in-the-loop approaches to enhance user oversight and decision-making in retrieval-based AI systems, and designing privacy-preserving retrieval techniques that safeguard user data while maintaining retrieval effectiveness.

By addressing these research directions, future work can contribute to the advancement of robust, interpretable, and fair embedding-based image retrieval systems. These efforts will be critical in ensuring that retrieval technologies are not only highly accurate and efficient but also trustworthy, transparent, and adaptable to real-world challenges. As retrieval models continue to evolve, interdisciplinary collaboration between AI researchers, domain experts, and policymakers will be essential in guiding the responsible deployment of these systems in healthcare, agriculture, and beyond.

Bibliography

[1] H. Sharma, "Machine learning in healthcare: Unpacking ethical and privacy concerns," *The Academic*, 2024. [Online]. Available: https://theacademic.in/wp-content/uploads/2024/09/71a.pdf

[2] S. Young, *Geoprocessing, Workflows in Remote Sensing Handbook, Volume II: Image Processing*. Google Books, 2024. [Online]. Available: https://books.google.com/books?hl=en&lr=&id=GxQuEQAAQBAJ&oi=fnd&pg=PA303

[3] E. M. O. Barmak, I. Krak, "Toward explainable deep learning in healthcare through transition matrix and user-friendly features," *Frontiers in Artificial Intelligence*, 2024. [Online]. Available: https://www.frontiersin.org/journals/artificial-intelligence/articles/10.3389/frai.2024.1482141/full

[4] e. a. L. Deka, A.V. Singh, "Automated diabetic retinopathy detection using multi-column deep neural networks," *IEEE Xplore*, 2024. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/10522355/

[5] S. P. K. P. V. Kulkarni, B. Nemade, "A short report on adhd detection using convolutional neural networks," *Frontiers in Psychiatry*, 2024. [Online]. Available: https://www.frontiersin.org/journals/psychiatry/articles/10.3389/fpsyt.2024.1426155/full

[6] M. U. T. Soni, D. Dupta, "Ensembled deep learning technique for cardiovascular disease detection: A cnn-gru approach," *IEEE 3rd World Conference on AI*, 2024. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/10730954/

[7] M. Ounissi, "Decoding the black box: Enhancing interpretability and trust in artificial intelligence for biomedical imaging," *Inria HAL*, 2024. [Online]. Available: https://inria.hal.science/tel-04786123/

[8] B. V. K.R. Arjun, K. Girish Kanavi, *Analysis of Biomedical Data with Explainable (XAI) and Responsive AI (RAI)*. Wiley Online Library, 2025. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/9781394302444.ch11

[9] F. Stieler, "The role of active learning in developing trustworthy ai-approaches for enhancing transparency and explainability in processes and systems," *University of Augsburg*, 2025.

[Online]. Available: https://opus.bibliothek.uni-augsburg.de/opus4/frontdoor/index/index/docId/117910

[10] N. Ghadge, "Machine learning: Enhancing intelligent search and information discovery," *SSRN Preprint*, 2024. [Online]. Available: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=4847981

[11] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004. [Online]. Available: https://www.cs.ubc.ca/~lowe/papers/ijcv04.pdf

[12] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2005, pp. 886–893. [Online]. Available: https://lear.inrialpes.fr/people/triggs/pubs/Dalal-cvpr05.pdf

[13] M. J. Swain and D. H. Ballard, "Color indexing," *International Journal of Computer Vision*, vol. 7, no. 1, pp. 11–32, 1991. [Online]. Available: https://link.springer.com/article/10.1007/BF00130487

[14] K. V. Nayak and J. S. Arunalatha, "Soft computing based artificial neural network and multiple feature set intelligence system for image retrieval," *Academia.edu*, 2023. [Online]. Available: https://www.researchgate.net/publication/357890978_Soft_Computing_based_Artificial_Neural_Network_and_Multiple_Feature_set_Intelligence_system_for_Image_Retrieval

[15] A. Mjahad, A. Polo-Aguado, and L. Llorens-Serrano, "Optimizing image feature extraction with convolutional neural networks for chicken meat detection applications," *Applied Sciences*, vol. 15, no. 2, 2025. [Online]. Available: https://www.mdpi.com/2076-3417/15/2/733

[16] F. Yang, N. A. Ismail, Y. Y. Pang, V. R. Kebande, and T. W. Koh, "A systematic literature review of deep learning approaches for sketch-based image retrieval: Datasets, metrics, and future directions," *IEEE Access*, vol. XX, pp. 1–XX, 2024. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/10413476/

[17] S. R. Dubey, "A decade survey of content based image retrieval using deep learning," *arXiv preprint arXiv:2012.00641*, 2020.

[18] W. Chen, Y. Liu, W. Wang, E. Bakker, T. Georgiou, P. Fieguth, L. Liu, and M. S. Lew, "Deep learning for instance retrieval: A survey," *arXiv preprint arXiv:2101.11282*, 2021.

[19] F. Yang and Others, "A comprehensive analysis on deep learning based image retrieval," in *Proceedings of the IEEE Conference on Image Processing*, 2019, accessed via IEEE Xplore. [Online]. Available: https://ieeexplore.ieee.org/document/10200622

[20] A. Riad, H. Elminir, and S. Abd-Elghany, "A literature review of image retrieval based on semantic concept," *International Journal of ...*, 2012. [Online]. Available: https://www.researchgate.net/profile/Sameh-Abd-El-Ghany-2/publication/258650984_A_Literature_Review_of_Image_Retrieval_based_On_Semantic_Concept/links/55e16b0008aecb1a7cc65e06/A-Literature-Review-of-Image-Retrieval-based-On-Semantic-Concept.pdf

[21] M. A. M. U. Toldo, M. and P. Zanuttigh, "Unsupervised domain adaptation in semantic segmentation: a review," *Technologies*, vol. 8, no. 2, 2020. [Online]. Available: https://www.mdpi.com/2227-7080/8/2/35

[22] S. Meduri, "The evolution of embeddings in machine learning," *International Journal of Computer Engineering Technology*, 2024. [Online]. Available: https://lib-index.com/index.php/IJCET/article/view/IJCET_15_04_068

[23] R. Miikkulainen, J. Liang, E. Meyerson, and A. Rawal, "Evolving deep neural networks," *Elsevier*, 2024. [Online]. Available: https://www.sciencedirect.com/science/article/pii/B9780323961042000026

[24] D. G. Lowe, "Object recognition from local scale-invariant features," in *Proceedings of the IEEE International Conference on Computer Vision*, 1999, pp. 1150–1157. [Online]. Available: https://www.cs.ubc.ca/~lowe/papers/iccv99.pdf

[25] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems*, 2012, pp. 1097–1105. [Online]. Available: https://papers.nips.cc/paper_files/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf

[26] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," https://arxiv.org/abs/1409.1556, 2014.

[27] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 1–9. [Online]. Available: https://www.cv-foundation.org/openaccess/content_cvpr_2015/papers/Szegedy_Going_Deeper_With_2015_CVPR_paper.pdf

[28] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp.

770–778. [Online]. Available: https://www.cv-foundation.org/openaccess/content_cvpr_2016/papers/He_Deep_Residual_Learning_CVPR_2016_paper.pdf

[29] L. van der Maaten and G. Hinton, "Visualizing data using t-sne," *Journal of Machine Learning Research*, vol. 9, pp. 2579–2605, 2008. [Online]. Available: https://www.jmlr.org/papers/volume9/vandermaaten08a/vandermaaten08a.pdf

[30] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, pp. 533–536, 1986. [Online]. Available: https://www.nature.com/articles/323533a0

[31] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, vol. 15, pp. 1929–1958, 2014. [Online]. Available: http://jmlr.org/papers/volume15/srivastava14a/srivastava14a.pdf

[32] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 4510–4520. [Online]. Available: https://openaccess.thecvf.com/content_cvpr_2018/papers/Sandler_MobileNetV2_Inverted_Residuals_CVPR_2018_paper.pdf

[33] M. Tan and Q. V. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," in *Proceedings of the International Conference on Machine Learning (ICML)*, 2019. [Online]. Available: https://proceedings.mlr.press/v97/tan19a/tan19a.pdf

[34] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," https://arxiv.org/abs/1312.6114, 2013.

[35] G. Koch, R. Zemel, and R. Salakhutdinov, "Siamese neural networks for one-shot image recognition," in *Proceedings of the International Conference on Machine Learning (ICML)*, 2015. [Online]. Available: https://www.cs.cmu.edu/~rsalakhu/papers/oneshot1.pdf

[36] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 815–823. [Online]. Available: https://www.cv-foundation.org/openaccess/content_cvpr_2015/papers/Schroff_FaceNet_A_Unified_2015_CVPR_paper.pdf

[37] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," arXiv preprint arXiv:1505.04597, 2015. [Online]. Available: https://arxiv.org/pdf/1505.04597.pdf

[38] G. Litjens, T. Kooi, B. E. Bejnordi, A. A. Setio, F. Ciompi, M. Ghafoorian, J. A. W. M. van der Laak, B. van Ginneken, and C. I. Sánchez, "A survey on deep learning in medical image analysis," *Medical Image Analysis*, vol. 42, pp. 60–88, 2017. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1361841517301135

[39] S. P. Mohanty, D. P. Hughes, and M. Salathé, "Using deep learning for image-based plant disease detection," *Frontiers in Plant Science*, vol. 7, p. 1419, 2016. [Online]. Available: https://www.frontiersin.org/articles/10.3389/fpls.2016.01419/full

[40] M. Bojarski, D. D. Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L. D. Jackel, M. Monfort, U. Muller, and J. Zhang, "End to end learning for self-driving cars," arXiv preprint arXiv:1604.07316, 2016.

[41] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009, pp. 248–255. [Online]. Available: https://ieeexplore.ieee.org/document/5206848

[42] T.-Y. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, C. L. Zitnick, and P. Dollár, "Microsoft coco: Common objects in context," in *European Conference on Computer Vision (ECCV)*, 2014, pp. 740–755. [Online]. Available: https://arxiv.org/abs/1405.0312

[43] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 2999–3007. [Online]. Available: https://openaccess.thecvf.com/content_ICCV_2017/papers/Lin_Focal_Loss_for_ICCV_2017_paper.pdf

[44] T. DeVries and G. W. Taylor, "Improved regularization of convolutional neural networks with cutout," arXiv preprint arXiv:1708.04552, 2017.

[45] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond empirical risk minimization," arXiv preprint arXiv:1710.09412, 2017.

[46] S. Imambi and K. B. Prakash, "Pytorch," in *Deep Learning with TensorFlow*. Springer, 2021, pp. 123–140. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-030-57077-4_10

[47] I. Vasilev, D. Slater, G. Spacagna, P. Roelants, and V. Zocca, *Python Deep Learning: Exploring deep learning techniques and neural network architectures with Pytorch, Keras, and TensorFlow*. Packt Publishing, 2019. [Online]. Available: https://books.google.com/books?hl=en&id=ESKEDwAAQBAJ

[48] A. F. Villán, *Mastering OpenCV 4 with Python: A practical guide covering topics from image processing, augmented reality to deep learning with OpenCV 4 and Python 3.7*. Packt Publishing, 2019. [Online]. Available: https://books.google.com/books?hl=en&id=w86PDwAAQBAJ

[49] Y. Mistry, D. Ingole, and M. Ingole, "Content based image retrieval using hybrid features and various distance metrics," *Journal of Electrical Systems and Information Technology*, vol. 5, no. 3, 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2314717216301155

[50] B. Xu, J. Bu, C. Chen, D. Cai, X. He, and W. Liu, "Efficient manifold ranking for image retrieval," *SIGIR Conference on Information Retrieval*, 2011. [Online]. Available: http://www.cad.zju.edu.cn/home/dengcai/Publication/Conference/2011_SIGIR-EMR.pdf

[51] M. Everingham, L. V. Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (voc) challenge," *International Journal of Computer Vision*, vol. 88, no. 2, pp. 303–338, 2010. [Online]. Available: https://link.springer.com/article/10.1007/s11263-009-0275-4

[52] I. T. Jolliffe, *Principal Component Analysis*. Springer, 2002. [Online]. Available: https://link.springer.com/book/10.1007/b98835

[53] N. Mehrabi, F. Morstatter, N. Saxena, K. P. Lerman, and A. Galstyan, "A survey on bias and fairness in machine learning," *ACM Computing Surveys*, vol. 54, no. 6, pp. 1–35, 2021. [Online]. Available: https://dl.acm.org/doi/10.1145/3457607

[54] J. Buolamwini and T. Gebru, "Gender shades: Intersectional accuracy disparities in commercial gender classification," in *Proceedings of the Conference on Fairness, Accountability, and Transparency*, 2018, pp. 77–91. [Online]. Available: https://proceedings.mlr.press/v81/buolamwini18a/buolamwini18a.pdf

[55] S. Xu, D. Hou, L. Pang, J. Deng, J. Xu, and H. Shen, "Invisible relevance bias: Text-image retrieval models prefer ai-generated images," in *ACM International Conference on Information Retrieval*. ACM, 2024. [Online]. Available: https://dl.acm.org/doi/abs/10.1145/3626772.3657750

[56] A. Mandal, S. Leavy, and S. Little, "Dataset diversity: Measuring and mitigating geographical bias in image search and retrieval," in *International Workshop on Trustworthy AI*. ACM, 2021. [Online]. Available: https://dl.acm.org/doi/abs/10.1145/3475731.3484956

[57] E. Ferrara, "Fairness and bias in artificial intelligence: A brief survey of sources, impacts, and mitigation strategies," *MDPI Science Journal*, vol. 6, no. 1, p. 3, 2023. [Online]. Available: https://www.mdpi.com/2413-4155/6/1/3

[58] F. Kong, S. Yuan, and W. Hao, "Mitigating test-time bias for fair image retrieval," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2024. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2023/file/e24570da4fa1c005b189104250993aee-Paper-Conference.pdf

[59] M. T. Ribeiro, S. Singh, and C. Guestrin, "Why should i trust you? explaining the predictions of any classifier," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016, pp. 1135–1144. [Online]. Available: https://dl.acm.org/doi/10.1145/2939672.2939778

[60] B. Kim, J. Park, and J. Suh, "Transparency and accountability in ai decision support: Explaining and visualizing convolutional neural networks for text information," *Decision Support Systems*, vol. 133, p. 113387, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167923620300579

[61] Z. Salahuddin, H. C. Woodruff, and A. Chatterjee, "Transparency of deep neural networks for medical image analysis: A review of interpretability methods," *Computers in Biology and Medicine*, vol. 141, p. 105090, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0010482521009057

[62] D. W. Joyce, A. Kormilitzin, K. A. Smith, and A. Cipriani, "Explainable artificial intelligence for mental health through transparency and interpretability for understandability," *npj Digital Medicine*, vol. 6, 2023. [Online]. Available: https://www.nature.com/articles/s41746-023-00751-9

[63] T. Räuker, A. Ho, and S. Casper, "Toward transparent ai: A survey on interpreting the inner structures of deep neural networks," in *IEEE Conference on AI Transparency*, 2023. [Online]. Available: https://arxiv.org/pdf/2207.13243

[64] D. S. Char, N. F. Shimada, and P. J. Shah, "Implementing artificial intelligence in health care—addressing ethical challenges," *Nature Medicine*, vol. 24, no. 10, pp. 1217–1220, 2018. [Online]. Available: https://pmc.ncbi.nlm.nih.gov/articles/PMC5962261/

[65] E. Topol, *Deep Medicine: How Artificial Intelligence Can Make Healthcare Human Again*. Basic Books, 2019. [Online]. Available: https://www.basicbooks.com/titles/eric-topol/deep-medicine/9781541644649/

[66] R. Shinde, S. Patil, K. Kotecha, and V. Potdar, "Securing ai-based healthcare systems using blockchain technology: A state-of-the-art systematic literature review and future research directions," *Transactions on Emerging Telecommunications Technologies*, 2024. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/ett.4884

[67] G. M. Minopoulos, V. A. Memos, and K. D. Stergiou, "A medical image visualization technique assisted with ai-based haptic feedback for robotic surgery and healthcare," *Applied Sciences*, vol. 13, no. 6, 2023. [Online]. Available: https://www.mdpi.com/2076-3417/13/6/3592

[68] I. Rahwan, A. Cebrian, F. Rahman, and et al., "Machine behaviour," *Nature*, vol. 568, pp. 477–486, 2019. [Online]. Available: https://www.nature.com/articles/s41586-019-1138-y

[69] T. Habuza, A. N. Navaz, F. Hashim, and F. Alnajjar, "Ai applications in robotics, diagnostic image analysis, and precision medicine: Current limitations, future trends, guidelines on cad systems for medicine," *Informatics in Medicine Unlocked*, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2352914821000861

[70] P. Manickam, S. A. Mariappan, S. M. Murugesan, and S. Hansda, "Artificial intelligence (ai) and internet of medical things (iomt) assisted biomedical systems for intelligent healthcare," *Biosensors*, vol. 12, no. 8, p. 562, 2022. [Online]. Available: https://www.mdpi.com/2079-6374/12/8/562

[71] M. Sani and W. Shah, "The pillars of ai safety: Integrating machine learning ethics and responsible fine-tuning," *ResearchGate*, 2024. [Online]. Available: https://www.researchgate.net/publication/387517359

[72] W. I. E. Jazibiyya, R. Dina, "Navigating the alignment challenges of diffusion models: Insights and innovations," *Preprints.org*, 2025. [Online]. Available: https://www.preprints.org/frontend/manuscript/0854b52b4974453bc59e98ed36b5683c/download_pub

[73] H. W. H. Muskan, "Advancing ai ethics: Responsible fine-tuning and safe deployment in artificial intelligence," *ResearchGate*, 2024. [Online]. Available: https://www.researchgate.net/publication/387517371

[74] V. L. R. DarBoux, "The ethical implications of machine intelligence in autonomous decision-making," *ResearchGate*, 2024. [Online]. Available: https://www.researchgate.net/publication/386071856

[75] F. R. C. A. Khodabakhshian, "Industry 5.0 in construction: Towards a more human-centric and ethical ai," *Springer*, 2025. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-031-77197-2_6

[76] P. H. G. Waters, W. Mapp, "Decisional value scores," *PhilPapers*, 2024. [Online]. Available: https://philpapers.org/rec/WATDVS

[77] P. W. L. Kester, M. Martens, "Addressing ethical challenges in automated vehicles: Bridging the gap with hybrid ai and augmented utilitarianism," *Springer*, 2024. [Online]. Available: https://link.springer.com/article/10.1007/s43681-024-00592-6

[78] M. C. Seixas, ""trustworthy ai": Contesting decisions taken by artificial intelligence in light of data protection," *UFBA Repository*, 2024. [Online]. Available: https://repositorio.ufba.br/handle/ri/41094

[79] R. Q. E. C. F. Colecchia, D. Giunchi, "Machine learning and immersive technologies for user-centered digital healthcare innovation," *Frontiers in Big Data*, 2025. [Online]. Available: https://www.frontiersin.org/journals/big-data/articles/10.3389/fdata.2025.1567941/full

[80] M. A.-L. CS Santos, "Externally validated and clinically useful machine learning algorithms to support patient-related decision-making in oncology: a scoping review," *BMC Medical Research*, 2025. [Online]. Available: https://pmc.ncbi.nlm.nih.gov/articles/PMC11843972/

[81] K. P. A. S. A Perivolaris, A Rueda, "Opinion: Mental health research: To augment or not to augment," *Frontiers in Psychiatry*, 2025. [Online]. Available: https://www.frontiersin.org/journals/psychiatry/articles/10.3389/fpsyt.2025.1539157/full

[82] K. Q. S Malik, LJ Frey, "Evaluating the predictive power of machine learning in cirrhosis mortality: a systematic review," *Journal of Medical Artificial Intelligence*, 2025. [Online]. Available: https://jmai.amegroups.org/article/view/9521/html

[83] R. Saxena, "Applications of natural language processing in the domain of mental health," *Authorea Preprints*, 2024. [Online]. Available: https://www.authorea.com/doi/full/10.36227/techrxiv.173014748.80471770

[84] E. A. R Alkawadri, JM Hillis, "Exploring the future of neurology: how ai is revolutionizing diagnoses, treatments, and beyond," *Frontiers in Neurology*, 2025. [Online]. Available: https://www.frontiersin.org/journals/neurology/articles/10.3389/fneur.2025.1556510/full

[85] B. K. H Jamalirad, M Jajroudi, "Ai predictive models and advancements in microdissection testicular sperm extraction for non-obstructive azoospermia: a systematic scoping review," *Human Reproduction Open*, 2024. [Online]. Available: https://academic.oup.com/hropen/article/doi/10.1093/hropen/hoae070/7906496

[86] "Iso/iec 25010:2011, systems and software engineering – systems and software quality requirements and evaluation (square) – system and software quality models," 2011, available online. [Online]. Available: https://www.iso.org/standard/35733.html

[87] R. F. A Banerjee, H Shan, "Artificial intelligence applications for cancer diagnosis in radiology," *Frontiers in Radiology*, 2025. [Online]. Available: https://www.frontiersin.org/journals/radiology/articles/10.3389/fradi.2025.1493783/full

[88] R. P. C Guimaraes, "Optimizing image logging acquisition with deep learning," *SPE Annual Technical Conference*, 2024. [Online]. Available: https://onepetro.org/SPEATCE/proceedings-abstract/24ATCE/24ATCE/563674

[89] S. C. A Tak, "Risk management in agile ai/ml projects: Identifying and mitigating data and model risks," *Journal of Technology and Systems*, 2024. [Online]. Available: https://ideas.repec.org/a/bhx/ojtjts/v6y2024i3p1-18id1824.html

[90] F. Westhäußer, "Robust and trustworthy deep learning-based disease detection and risk assessment on mri and histopathological images," *ProQuest*, 2024. [Online]. Available: https://search.proquest.com/openview/9edf35810f70558c13b1e1a1997d7667/1?pq-origsite=gscholar

[91] J. Wan, D. Wang, S. C. H. Hoi, P. Wu, and J. Zhu, "Deep learning for content-based image retrieval: A comprehensive study," in *Proceedings of the 22nd ACM International Conference on Multimedia*. ACM, 2014, pp. 157–166. [Online]. Available: https://dl.acm.org/doi/abs/10.1145/2647868.2654948

[92] N. Passalis, A. Iosifidis, M. Gabbouj, and A. Tefas, "Variance-preserving deep metric learning for content-based image retrieval," *Pattern Recognition Letters*, vol. 131, pp. 269–275, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167865519303629

[93] S. Gkelios, A. Sophokleous, S. Plakias, and Y. Boutalis, "Deep convolutional features for image retrieval," *Expert Systems with Applications*, vol. 184, p. 115654, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S095741742100381X

[94] W. Chen, Y. Liu, W. Wang, and E. M. Bakker, "Deep learning for instance retrieval: A survey," *IEEE Transactions on Neural Networks and Learning Systems*, 2022. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/9933854/

[95] K. L. D. Dataset, "Leaf diseases dataset," https://www.kaggle.com/datasets/zaneneave/plant-diseases, 2020, accessed: 2025-02-22.

[96] K. B. T. M. Dataset, "Brain tumour mri dataset," https://www.kaggle.com/datasets/fernando2rad/brain-tumor-mri-images-44c, 2020, accessed: 2025-02-22.

[97] K. C.-. D. X. rays Forgery Dataset, "Covid-19 digital x-rays forgery dataset," https://www.kaggle.com/datasets/nourmahmoud/covid19-digital-xrays-forgery-dataset, 2020, accessed: 2025-02-22.

[98] K. P. I. C. Dataset, "Pokémon image classification dataset," https://www.kaggle.com/datasets/lantian773030/pokemonclassification, 2020, accessed: 2025-02-22.

[99] C. Sun, A. Shrivastava, S. Singh, and A. Gupta, "Revisiting unreasonable effectiveness of data in deep learning era," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 843–852. [Online]. Available: https://openaccess.thecvf.com/content_iccv_2017/html/Sun_Revisiting_Unreasonable_Effectiveness_ICCV_2017_paper.html

[100] D. Mahajan, R. B. Girshick, V. Ramanathan, K. He, M. Paluri, Y. Li, A. Bharambe, and L. van der Maaten, "Exploring the limits of weakly supervised pretraining," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 181–196. [Online]. Available: https://arxiv.org/abs/1805.00932

[101] B. Zhou, A. Lapedriza, A. Khosla, A. Oliva, and A. Torralba, "Places: A 10 million image database for scene recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, no. 10, pp. 2490–2504, 2020. [Online]. Available: https://arxiv.org/abs/1610.02055

[102] B. Recht, R. Roelofs, L. Schmidt, and V. Shankar, "Do imagenet classifiers generalize to imagenet?" in *Proceedings of the 36th International Conference on Machine Learning (ICML)*, 2019, pp. 5389–5400. [Online]. Available: https://proceedings.mlr.press/v97/recht19a/recht19a.pdf

[103] M. Buda, A. Maki, and M. Mazurowski, "A systematic study of the class imbalance problem in convolutional neural networks," *Neural Networks*, vol. 106, pp. 249–259, 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0893608018302107

[104] G. Wilson and D. J. Cook, "A survey of unsupervised deep domain adaptation," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 11, no. 5, pp. 1–46, 2020. [Online]. Available: https://dl.acm.org/doi/10.1145/3400066

[105] A. Kolesnikov, L. Beyer, X. Zhai, J. Puigcerver, J. Yung, and A. Dosovitskiy, "Big transfer (bit): General visual representation learning," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020, pp. 491–507. [Online]. Available: https://arxiv.org/pdf/1912.11370.pdf

[106] J. Buolamwini and T. Gebru, "Gender shades: Intersectional accuracy disparities in commercial gender classification," in *Proceedings of the Conference on Fairness, Accountability, and Transparency*, 2018, pp. 77–91. [Online]. Available: https://proceedings.mlr.press/v81/buolamwini18a/buolamwini18a.pdf

[107] L. Dou, Z. Zhang, L. Xu, and Q. Zou, "A novel computational tool for imbalance classification of human nonhistone crotonylation sites based on convolutional neural networks with focal loss," *Computational and Structural Biotechnology Journal*, vol. 20, pp. 2461–2473, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S200103702200246X

[108] X. Deng, Y. Xu, L. Chen, W. Zhong, and A. Jolfaei, "Dynamic clustering method for imbalanced learning based on adaboost," *The Journal of Supercomputing*, vol. 76, no. 5, pp. 3175–3198, 2020. [Online]. Available: https://link.springer.com/article/10.1007/s11227-020-03211-3

[109] G. Idakwo, S. Thangapandian, J. Luttrell, and Y. Li, "Structure–activity relationship-based chemical classification of highly imbalanced tox21 datasets," *Journal of Cheminformatics*, vol. 12, no. 1, p. 68, 2020. [Online]. Available: https://link.springer.com/article/10.1186/s13321-020-00468-x

[110] M. Chabbouh, S. Bechikh, and E. Mezura-Montes, "Evolutionary optimization of the area under precision-recall curve for classifying imbalanced multi-class data," *Journal of Heuristic Methods*, 2025. [Online]. Available: https://link.springer.com/article/10.1007/s10732-024-09544-z

[111] D. Dablain and B. Krawczyk, "Deepsmote: Fusing deep learning and smote for imbalanced data," *IEEE Transactions on Neural Networks and Learning Systems*, 2022. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/9694621/

[112] S. Agarwal, S. Muku, and S. Anand, "Does data repair lead to fair models? curating contextually fair data to reduce model bias," in *Proceedings of the IEEE Winter*

Conference on Applications of Computer Vision (WACV). IEEE, 2022. [Online]. Available: http://openaccess.thecvf.com/content/WACV2022/html/Agarwal_Does_Data_Repair_Lead_to_Fair_Models_Curating_Contextually_Fair_WACV_2022_paper.html

[113] V. Sehwag, R. Oak, M. Chiang, and P. Mittal, "Time for a background check! uncovering the impact of background features on deep neural networks," *arXiv preprint arXiv:2006.14077*, 2020. [Online]. Available: https://arxiv.org/abs/2006.14077

[114] J. G. A. Barbedo, "Impact of dataset size and variety on the effectiveness of deep learning and transfer learning for plant disease classification," *Computers and Electronics in Agriculture*, vol. 153, pp. 46–53, 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0168169918304617

[115] A. R. Luca, T. F. Ursuleanu, L. Gheorghe, and M. P. Bogdan, "Impact of quality, type, and volume of data used by deep learning models in the analysis of medical images," *Informatics in Medicine Unlocked*, vol. 29, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2352914822000612

[116] S. Arif and F. Shafait, "Table detection in document images using foreground and background features," in *Proceedings of the International Conference on Digital Image Computing: Techniques and Applications (DICTA)*. IEEE, 2018. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/8615795/

[117] S. Dodge and L. Karam, "Understanding how image quality affects deep neural networks," in *Proceedings of the 2016 International Conference on Quality of Multimedia Experience (QoMEX)*. IEEE, 2016. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/7498955/

[118] R. A. Sobbahi and J. Tekli, "Comparing deep learning models for low-light natural scene image enhancement and their impact on object detection and classification," *Signal Processing: Image Communication*, vol. 109, pp. 1163–1175, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S092359652200131X

[119] S. Arif and F. Shafait, "Foreground and background modeling for document layout analysis," in *Proceedings of the International Conference on Frontiers in Handwriting Recognition (ICFHR)*. IEEE, 2018. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/8615795/

[120] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, 2004. [Online]. Available: https://ieeexplore.ieee.org/document/1284395

[121] A. Kirillov, E. Mintun, N. Naik, H. Pirsiavash, R. Girshick, and P. Dollár, "Segment anything," https://segment-anything.com/, 2023, accessed: 2025-02-22.

[122] M. L. Community, "Segment anything model (sam) benchmark on 22 consumer gpus," 2024. [Online]. Available: https://www.reddit.com/r/MachineLearning/comments/1agr8rm/p_segment_anything_model_sam_benchmark_on_22/

[123] L. AI, "Segment anything model and friends," 2024. [Online]. Available: https://www.lightly.ai/post/segment-anything-model-and-friends

[124] X. Qin, Z. Zhang, Z. Shen, X. Tan, and C. Zhang, "U-2-net: Going deeper with nested u-structure for salient object detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 7450–7459. [Online]. Available: https://www.scilit.com/publications/7ca9f4651155cc17bfc5d07fb1344c99

[125] H. Wan, X. Zeng, Z. Fan, S. Zhang, and M. Kang, "U2espnet—a lightweight and high-accuracy convolutional neural network for real-time semantic segmentation of visible branches," *Computers and Electronics in Agriculture*, 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S016816992200850X

[126] W. Fang, Y. Fu, and V. Sheng, "Fps-u2net: Combining u2net and multi-level aggregation architecture for fire point segmentation in remote sensing images," *Computers Geosciences*, 2024. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0098300424001110

[127] L. Chen, G. Papandreou, F. Schroff, and H. Adam, "Encoder-decoder with atrous separable convolution for semantic image segmentation," *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018. [Online]. Available: https://arxiv.org/abs/1802.02611

[128] Y. Wang, L. Yang, X. Liu, and P. Yan, "An improved semantic segmentation algorithm for high-resolution remote sensing images based on deeplabv3+," *Scientific Reports*, 2024. [Online]. Available: https://www.nature.com/articles/s41598-024-60375-1

[129] D. Yi, N. Li, H. Chen, Z. Sun, and J. Gao, "Real-time semantic segmentation of solar photovoltaic arrays for autonomous uav flights," *IEEE Xplore*, 2024. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/10662016/

[130] T. Liu, W. Chen, X. Lin, Y. Mu, J. Huang, D. Gao, and J. Xu, "Defogging learning based on an improved deeplabv3+ model for accurate foggy forest fire segmentation," *Forests*, 2023. [Online]. Available: https://www.mdpi.com/1999-4907/14/9/1859

[131] Z. Guo, K. Liu, X. Sun, C. Ding, and S. Li, "An overlay accelerator of deeplab cnn for spacecraft image segmentation on fpga," *Remote Sensing*, 2024. [Online]. Available: https://www.mdpi.com/2072-4292/16/5/894

[132] H. Jung, J. Suhr, and Y. Lee, "Semantic segmentation network slimming and edge deployment for real-time forest fire or flood monitoring systems using unmanned aerial vehicles," *Electronics*, 2023. [Online]. Available: https://www.mdpi.com/2079-9292/12/23/4795

[133] M. Lee and H. J. Mun, "Comparison analysis and case study for deep learning-based object detection algorithm," *International Journal of Advanced Science and Convergence (IJASC)*, vol. 2, no. 4, pp. 21–34, 2020. [Online]. Available: https://journalofbigdata.springeropen.com/articles/10.1186/s40537-021-00434-w

[134] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "Tensorflow: Large-scale machine learning on heterogeneous distributed systems," in *Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI '16)*, 2016, pp. 265–283. [Online]. Available: https://arxiv.org/pdf/1603.04467.pdf

[135] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2019, pp. 8026–8037. [Online]. Available: https://arxiv.org/pdf/1912.01703.pdf

[136] F. Chollet, "Keras," https://keras.io, 2015, accessed: 2025-02-22.

[137] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998. [Online]. Available: https://ieeexplore.ieee.org/document/726791

[138] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," in *Journal of Machine Learning Research*, vol. 15, 2014, pp. 1929–1958. [Online]. Available: http://jmlr.org/papers/volume15/srivastava14a/srivastava14a.pdf

[139] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 1026–1034. [Online]. Available: https://openaccess.thecvf.com/content_iccv_2015/papers/He_Delving_Deep_into_ICCV_2015_paper.pdf

[140] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proceedings of the 32nd International Conference on Machine Learning (ICML)*, 2015, pp. 448–456. [Online]. Available: https://proceedings.mlr.press/v37/ioffe15.pdf

[141] A. Name, "An illustration of resnet-50 layers architecture," in *ResearchGate*, 2019, available at: https://www.researchgate.net/figure/An-illustration-of-ResNet-50-layers-architecture_fig1_350421671.

[142] M. Ibrahim, "The basics of resnet50," https://wandb.ai/mostafaibrahim17/ml-articles/reports/The-Basics-of-ResNet50---Vmlldzo2NDkwNDE2, 2020, accessed: 2025-02-22.

[143] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," https://arxiv.org/abs/1704.04861, 2017, accessed: 2025-02-22.

[144] R. User, "The architecture of the mobilenetv2 network," https://www.researchgate.net/figure/The-architecture-of-the-MobileNetv2-network_fig3_342856036, 2019, accessed: 2025-02-22.

[145] ——, "The detailed architecture of efficientnet-b0," https://www.researchgate.net/publication/361521546/figure/fig2/AS:11431281122798843@1677553158796/The-detailed-architecture-of-EfficientNet-B0-EfficientNet-B0-consists-of-seven-blocks.png, 2022, accessed: 2025-02-22.

[146] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. [Online].

Available: https://www.cv-foundation.org/openaccess/content_cvpr_2015/html/Szegedy_Going_Deeper_With_2015_CVPR_paper.html

[147] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014. [Online]. Available: https://arxiv.org/abs/1409.1556

[148] A. Alzu'bi, A. Amira, and N. Ramzan, "Content-based image retrieval with compact deep convolutional features," *Neurocomputing*, 2017. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0925231217306185

[149] S. Kumar, M. K. Singh, and M. Mishra, "Efficient deep feature-based semantic image retrieval," *Neural Processing Letters*, 2023. [Online]. Available: https://link.springer.com/article/10.1007/s11063-022-11079-y

[150] G. Huang, Z. Liu, L. V. D. Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. [Online]. Available: https://arxiv.org/abs/1608.06993

[151] T. Cover and P. Hart, "Nearest neighbor pattern classification," *IEEE Transactions on Information Theory*, vol. 13, no. 1, pp. 21–27, 1967. [Online]. Available: https://ieeexplore.ieee.org/document/1053964

[152] N. S. Altman, "An introduction to kernel and nearest-neighbor nonparametric regression," *The American Statistician*, vol. 46, no. 3, pp. 175–185, 1992. [Online]. Available: https://www.jstor.org/stable/2685209

[153] K. Beyer, J. Goldstein, R. Ramakrishnan, and U. Shaft, "When is "nearest neighbor" meaningful?" in *Proceedings of the 7th International Conference on Database Theory (ICDT)*, 1999, pp. 217–235. [Online]. Available: https://link.springer.com/chapter/10.1007/3-540-49257-7_15

[154] J. Johnson, M. Douze, and H. Jégou, "Billion-scale similarity search with gpus," in *arXiv preprint arXiv:1702.08734*, 2017.

[155] H. Jégou, M. Douze, and C. Schmid, "Product quantization for nearest neighbor search," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2011, pp. 604–613. [Online]. Available: https://ieeexplore.ieee.org/document/5432202

[156] E. Bernard *et al.*, "Annoy: Approximate nearest neighbors in c++/python," 2014, available at https://github.com/spotify/annoy.

[157] M. Muja and D. G. Lowe, "Fast approximate nearest neighbors with automatic algorithm configuration," in *VISAPP (1)*, 2009, pp. 331–340. [Online]. Available: https://www.cs.ubc.ca/research/flann/uploads/FLANN/flann_visapp09.pdf

[158] P. Indyk and R. Motwani, "Approximate nearest neighbors: Towards removing the curse of dimensionality," in *Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing (STOC)*, 1998, pp. 604–613. [Online]. Available: https://dl.acm.org/doi/10.1145/276698.276876

[159] W. Dong, M. Charikar, and K. Li, "Efficient k-nearest neighbor graph construction for generic similarity measures," in *Proceedings of the 20th International Conference on World Wide Web (WWW)*, 2011, pp. 211–220. [Online]. Available: https://www.semanticscholar.org/paper/Efficient-k-nearest-neighbor-graph-construction-for-Dong-Charikar/f17c6e164ccc7ec1ad91b3fbbafe8f84664e9803

[160] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016. [Online]. Available: https://www.deeplearningbook.org/

[161] M. Zaharia, R. Xin, P. Wendell, T. Das, M. Armbrust, A. Dave, X. Meng, J. Rosen, S. Venkataraman, M. J. Franklin, A. Ghodsi, J. Gonzalez, and S. Shenker, "Apache spark: Unified analytics engine for big data," 2016, available at: https://spark.apache.org/.

[162] J. Dean, "Large scale distributed systems," 2012, available online, accessed 2025-02-22. [Online]. Available: https://research.google/pubs/pub38134/

[163] L. Liao, H. Li, W. Shang, and L. Ma, "An empirical study of the impact of hyperparameter tuning and model optimization on the performance properties of deep neural networks," *ACM Transactions on Software Engineering and Methodology*, 2022. [Online]. Available: https://dl.acm.org/doi/abs/10.1145/3506695

[164] N. DeCastro-García and L. M. Castañeda, "Effect of dataset sampling in the hyperparameter optimization phase over machine learning efficiency," *Wiley Online Library*, 2019. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1155/2019/6278908

[165] S. R. Young, D. C. Rose, T. P. Karnowski, and S. H. Lim, "Optimizing deep learning hyper-parameters through an evolutionary algorithm," *Proceedings of the Workshop on Machine Learning in High-Performance Computing Environments*, 2015. [Online]. Available: https://dl.acm.org/doi/abs/10.1145/2834892.2834896

[166] D. Preuveneers, I. Tsingenopoulos, and W. Joosen, "Resource usage and performance trade-offs for machine learning models in smart environments," *Sensors*, 2020. [Online]. Available: https://www.mdpi.com/1424-8220/20/4/1176

[167] P. S. Pravin, J. Z. M. Tan, K. S. Yap, and Z. Wu, "Hyperparameter optimization strategies for stochastic energy-efficient scheduling," *Digital Chemical Engineering*, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2772508122000370

[168] Y. A. Ali, E. M. Awwad, M. Al-Razgan, and A. Maarouf, "Hyperparameter search for machine learning algorithms for optimizing computational complexity," *Processes*, 2023. [Online]. Available: https://www.mdpi.com/2227-9717/11/2/349

[169] N. Bacanin, C. Stoean, M. Zivkovic, and M. Rakic, "On the benefits of using metaheuristics in the hyperparameter tuning of deep learning models," *Energies*, 2023. [Online]. Available: https://www.mdpi.com/1996-1073/16/3/1434

[170] L. Wu, G. Perin, and S. Picek, "Reinforcement learning for hyperparameter tuning in deep learning-based side-channel analysis," *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2021. [Online]. Available: https://tches.iacr.org/index.php/TCHES/article/view/8989

[171] M. Lin, Q. Chen, and S. Yan, "Network in network," 2013. [Online]. Available: https://arxiv.org/abs/1312.4400

[172] X. Yin, L. Wang, W. Jia, and C. Jin, "Semi-supervised transformation and deep embedding-based anomaly identification for agricultural internet of things," *IEEE Sensors Journal*, 2020. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/9309393/

[173] J. Ding, B. Li, C. Xu, Y. Qiao, and L. Zhang, "Diagnosing crop diseases based on domain-adaptive pre-training bert of electronic medical records," *Applied Intelligence*, 2023. [Online]. Available: https://link.springer.com/article/10.1007/s10489-022-04346-x

[174] R. Fernandes, A. Pessoa, M. Salgado, and A. D. Paiva, "Enhancing image annotation with object tracking and image retrieval: A systematic review," *IEEE Transactions on AI*, 2024. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/10540016/

[175] B. Ionescu, M. Lupu, M. Rohm, and A. L. Gînsca, "Datasets column: diversity and credibility for social images and image retrieval," *ACM SIGMultimedia*, 2018. [Online]. Available: https://dl.acm.org/doi/fullHtml/10.1145/3178422.3178429

[176] A. Gordo, J. Almazán, J. Revaud, and D. Larlus, "Deep image retrieval: Learning global representations for image search," in *Computer Vision–ECCV 2016*. Springer, 2016. [Online]. Available: https://arxiv.org/pdf/1604.01325

[177] W. Chen, Y. Liu, W. Wang, and E. M. Bakker, "Deep learning for instance retrieval: A survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/9933854/

[178] D. Bhati, F. Neha, and M. Amiruzzaman, "A survey on explainable artificial intelligence (xai) techniques for visualizing deep learning models in medical imaging," *Journal of Imaging*, vol. 10, no. 10, p. 239, 2024. [Online]. Available: https://www.mdpi.com/2313-433X/10/10/239

[179] B. Van der Velden, H. Kuijf, and K. Gilhuijs, "Explainable artificial intelligence (xai) in deep learning-based medical image analysis," *Medical Image Analysis*, vol. 78, p. 102409, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1361841522001177

[180] A. Singh, S. Sengupta, and V. Lakshminarayanan, "Explainable deep learning models in medical image analysis," *Journal of Imaging*, vol. 6, no. 6, p. 52, 2020. [Online]. Available: https://www.mdpi.com/2313-433X/6/6/52

[181] B. Hu, B. Vasu, and A. Hoogs, "X-mir: Explainable medical image retrieval," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 2022. [Online]. Available: https://openaccess.thecvf.com/content/WACV2022/papers/Hu_X-MIR_EXplainable_Medical_Image_Retrieval_WACV_2022_paper.pdf

[182] A. Das and P. Rad, "Opportunities and challenges in explainable artificial intelligence (xai): A survey," *arXiv preprint*, 2020. [Online]. Available: https://arxiv.org/pdf/2006.11371

[183] N. Gozzi, E. Giacomello, M. Sollini, and M. Kirienko, "Image embeddings extracted from cnns outperform other transfer learning approaches in classification of chest radiographs," *Diagnostics*, vol. 12, no. 9, p. 2084, 2022. [Online]. Available: https://www.mdpi.com/2075-4418/12/9/2084

Chapter A

Domain-Specific Retrieval Pipeline (DSRP)

A.0.1 Leaf Disease Dataset Tables

Table A.1: Retrieval Results for Leaf Disease Dataset

		KNN							FAISS						
Model	Background	Top 10 Acc.	P@5	P@10	R@5	R@10	F1@5	F1@10	Top 10 Acc.	P@5	P@10	R@5	R@10	F1@5	F1@10
Lightweight	Yes	0.98	0.89	0.89	0.97	0.98	0.91	0.91	0.98	0.89	0.88	0.96	0.98	0.92	0.93
Moderate	Yes	0.99	0.98	0.98	0.99	0.99	0.99	0.99	0.99	0.98	0.98	0.99	0.99	0.98	0.98
Heavy	Yes	0.94	0.83	0.82	0.92	0.94	0.86	0.85	0.94	0.84	0.83	0.93	0.94	0.88	0.88
ResNet50	Yes	0.99	0.94	0.92	0.99	0.99	0.96	0.95	0.99	0.94	0.92	0.99	0.99	0.96	0.95
MobileNetV2	Yes	0.99	0.89	0.87	0.99	0.99	0.92	0.91	0.99	0.91	0.88	0.99	0.99	0.95	0.93
EfficientNetB0	Yes	0.99	0.92	0.90	0.99	0.99	0.94	0.93	0.99	0.92	0.90	0.99	0.99	0.95	0.94
Lightweight	No	0.96	0.82	0.81	0.94	0.96	0.85	0.85	0.96	0.82	0.82	0.94	0.96	0.88	0.88
Moderate	No	0.99	0.97	0.97	0.99	0.99	0.98	0.98	0.99	0.97	0.97	0.99	0.99	0.98	0.98
Heavy	No	0.99	0.96	0.96	0.99	0.99	0.97	0.97	0.99	0.96	0.95	0.99	0.99	0.97	0.97
ResNet50	No	0.99	0.90	0.87	0.99	0.99	0.92	0.91	0.99	0.90	0.88	0.99	0.99	0.94	0.93
MobileNetV2	No	0.99	0.86	0.83	0.98	0.99	0.89	0.88	0.99	0.87	0.84	0.98	0.99	0.92	0.91
EfficientNetB0	No	0.99	0.88	0.85	0.98	0.99	0.91	0.90	0.99	0.89	0.86	0.98	0.99	0.93	0.92

Table A.2: Inference Results for Leaf Disease Dataset

Model	Background	KNN Inference Time (s)	FAISS Inference Time (s)
Lightweight	Yes	0.0806	0.0665
Lightweight	No	0.0807	0.0711
Moderate	Yes	0.0866	0.0729
Moderate	No	0.0855	0.0821
Heavy	Yes	0.1125	0.0874
Heavy	No	0.1145	0.0959
ResNet50	Yes	0.2394	0.1290
ResNet50	No	0.2554	0.1383
MobileNetV2	Yes	0.1737	0.0848
MobileNetV2	No	0.1805	0.0878
EfficientNetB0	Yes	0.1753	0.0931
EfficientNetB0	No	0.1871	0.0968

A.0.2 Brain Tumour MRI Scans Dataset Tables

Table A.3: Retrieval Results for Brain Tumour MRI Dataset

		KNN							FAISS						
Model	Background	Top 10 Acc.	P@5	P@10	R@5	R@10	F1@5	F1@10	Top 10 Acc.	P@5	P@10	R@5	R@10	F1@5	F1@10
Lightweight	Yes	0.85	0.48	0.44	0.77	0.85	0.55	0.53	0.85	0.49	0.45	0.76	0.85	0.60	0.59
Moderate	Yes	0.92	0.80	0.80	0.89	0.92	0.83	0.82	0.91	0.82	0.81	0.89	0.91	0.85	0.86
Heavy	Yes	0.88	0.83	0.83	0.87	0.88	0.84	0.84	0.87	0.84	0.84	0.87	0.87	0.85	0.85
ResNet50	Yes	0.97	0.67	0.56	0.95	0.97	0.75	0.66	0.97	0.67	0.57	0.95	0.97	0.79	0.72
MobileNetV2	Yes	0.96	0.66	0.56	0.94	0.96	0.74	0.65	0.96	0.68	0.56	0.95	0.96	0.79	0.71
EfficientNetB0	Yes	0.97	0.66	0.55	0.96	0.97	0.74	0.65	0.97	0.67	0.56	0.95	0.97	0.79	0.71

Table A.4: Inference Results for Brain Tumour MRI Dataset

Model	Background	KNN Inference Time (s)	FAISS Inference Time (s)
Lightweight	Yes	0.1006	0.0881
Moderate	Yes	0.0988	0.0730
Heavy	Yes	0.1207	0.0875
ResNet50	Yes	0.1701	0.1015
MobileNetV2	Yes	0.1285	0.0716
EfficientNetB0	Yes	0.1194	0.0852

A.0.3 COVID-19 Scans Dataset Tables

Table A.5: Retrieval Results for COVID-19 Scans Dataset

		KNN							FAISS						
Model	Background	Top 10 Acc.	P@5	P@10	R@5	R@10	F1@5	F1@10	Top 10 Acc.	P@5	P@10	R@5	R@10	F1@5	F1@10
Lightweight	Yes	0.93	0.38	0.38	0.82	0.93	0.49	0.51	0.93	0.39	0.38	0.83	0.93	0.53	0.54
Moderate	Yes	0.93	0.62	0.62	0.89	0.93	0.70	0.71	0.93	0.62	0.62	0.89	0.93	0.73	0.74
Heavy	Yes	0.92	0.58	0.58	0.87	0.92	0.67	0.67	0.92	0.59	0.58	0.86	0.92	0.70	0.71
ResNet50	Yes	0.97	0.36	0.36	0.87	0.97	0.49	0.51	0.96	0.36	0.36	0.87	0.96	0.51	0.52
MobileNetV2	Yes	0.96	0.36	0.36	0.86	0.96	0.48	0.49	0.96	0.36	0.36	0.86	0.96	0.51	0.52
EfficientNetB0	Yes	0.95	0.37	0.37	0.86	0.95	0.49	0.51	0.95	0.37	0.37	0.86	0.95	0.52	0.53

Table A.6: Inference Results for COVID-19 Scans Dataset

Model	Background	KNN Inference Time (s)	FAISS Inference Time (s)
Lightweight	Yes	0.0676	0.0577
Moderate	Yes	0.0678	0.0630
Heavy	Yes	0.0868	0.0831
ResNet50	Yes	0.1783	0.1013
MobileNetV2	Yes	0.1123	0.0681
EfficientNetB0	Yes	0.1184	0.0754

A.0.4 Pokémon Dataset Tables

Table A.7: Retrieval Results for Pokémon Dataset

		KNN							FAISS						
Model	Background	Top 10 Acc.	P@5	P@10	R@5	R@10	F1@5	F1@10	Top 10 Acc.	P@5	P@10	R@5	R@10	F1@5	F1@10
Lightweight	Yes	0.62	0.26	0.23	0.52	0.62	0.32	0.30	0.62	0.27	0.25	0.52	0.62	0.36	0.36
Moderate	Yes	0.70	0.45	0.44	0.64	0.70	0.50	0.49	0.70	0.48	0.47	0.64	0.70	0.55	0.56
Heavy	Yes	0.70	0.66	0.66	0.69	0.70	0.67	0.67	0.69	0.68	0.68	0.69	0.69	0.68	0.68
ResNet50	Yes	0.82	0.35	0.26	0.72	0.82	0.45	0.37	0.83	0.38	0.29	0.77	0.83	0.51	0.43
MobileNetV2	Yes	0.81	0.38	0.29	0.73	0.81	0.47	0.39	0.83	0.40	0.30	0.75	0.83	0.52	0.44
EfficientNetB0	Yes	0.82	0.44	0.34	0.75	0.82	0.52	0.44	0.85	0.48	0.38	0.78	0.85	0.59	0.53
Lightweight	No	0.67	0.28	0.25	0.56	0.67	0.34	0.33	0.67	0.30	0.27	0.57	0.67	0.39	0.38
Moderate	No	0.69	0.48	0.45	0.64	0.69	0.52	0.50	0.69	0.50	0.50	0.64	0.69	0.56	0.58
Heavy	No	0.71	0.62	0.62	0.68	0.71	0.63	0.63	0.67	0.64	0.64	0.67	0.67	0.65	0.65
ResNet50	No	0.88	0.51	0.40	0.83	0.88	0.59	0.51	0.89	0.55	0.45	0.85	0.89	0.67	0.60
MobileNetV2	No	0.85	0.46	0.36	0.80	0.85	0.55	0.46	0.87	0.48	0.38	0.81	0.87	0.60	0.53
EfficientNetB0	No	0.89	0.55	0.45	0.85	0.89	0.63	0.55	0.90	0.58	0.48	0.86	0.90	0.69	0.63

Table A.8: Inference Results for Pokémon Dataset

Model	Background	KNN Inference Time (s)	FAISS Inference Time (s)
Lightweight	Yes	0.0754	0.0667
Lightweight	No	0.1128	0.0633
Moderate	Yes	0.0821	0.0721
Moderate	No	0.1480	0.0745
Heavy	Yes	0.1024	0.0905
Heavy	No	0.1169	0.0903
ResNet50	Yes	0.1525	0.1116
ResNet50	No	0.1633	0.1118
MobileNetV2	Yes	0.1178	0.1227
MobileNetV2	No	0.1154	0.0726
EfficientNetB0	Yes	0.1271	0.1512
EfficientNetB0	No	0.1204	0.0803

Chapter B

Pre-trained Feature Extraction Pipeline (PMRP)

B.0.1 Leaf Disease Dataset Tables

Table B.1: Retrieval Results for Leaf Disease Dataset

		KNN							FAISS						
Model	Background	Top 10 Acc.	P@5	P@10	R@5	R@10	F1@5	F1@10	Top 10 Acc.	P@5	P@10	R@5	R@10	F1@5	F1@10
ResNet50	Yes	0.99	0.94	0.92	0.99	0.99	0.96	0.95	0.99	0.94	0.92	0.99	0.99	0.96	0.95
MobileNetV2	Yes	0.99	0.89	0.87	0.99	0.99	0.92	0.91	0.99	0.91	0.88	0.99	0.99	0.93	0.92
EfficientNetB0	Yes	0.99	0.92	0.90	0.99	0.99	0.94	0.93	0.99	0.92	0.90	0.99	0.99	0.94	0.93
ResNet50	No	0.99	0.90	0.87	0.99	0.99	0.92	0.91	0.99	0.90	0.88	0.99	0.99	0.93	0.91
MobileNetV2	No	0.98	0.86	0.83	0.98	0.99	0.89	0.88	0.98	0.87	0.84	0.98	0.99	0.90	0.89
EfficientNetB0	No	0.99	0.88	0.85	0.98	0.99	0.91	0.90	0.99	0.89	0.86	0.98	0.99	0.92	0.90

Table B.2: Inference Results for Leaf Disease Dataset

Model	Background	KNN Inference Time (s)	FAISS Inference Time (s)
ResNet50	Yes	0.1180	0.1197
ResNet50	No	0.1307	0.1041
MobileNetV2	Yes	0.0877	0.0716
MobileNetV2	No	0.0917	0.0741
EfficientNetB0	Yes	0.0972	0.0823
EfficientNetB0	No	0.1067	0.0848

B.0.2 Brain Tumour MRI Scans Dataset Tables

Table B.3: Retrieval Results for Brain Tumour MRI Scans Dataset

		KNN							FAISS						
Model	Background	Top 10 Acc.	P@5	P@10	R@5	R@10	F1@5	F1@10	Top 10 Acc.	P@5	P@10	R@5	R@10	F1@5	F1@10
ResNet50	Yes	0.97	0.67	0.56	0.95	0.97	0.77	0.66	0.97	0.67	0.57	0.95	0.97	0.75	0.67
MobileNetV2	Yes	0.96	0.66	0.56	0.94	0.96	0.74	0.65	0.96	0.67	0.56	0.95	0.96	0.75	0.66
EfficientNetB0	Yes	0.97	0.66	0.55	0.96	0.97	0.74	0.65	0.97	0.67	0.56	0.95	0.97	0.75	0.66

Table B.4: Inference Results for Brain Tumour MRI Scans Dataset

Model	Background	KNN Inference Time (s)	FAISS Inference Time (s)
ResNet50	Yes	0.1201	0.1055
MobileNetV2	Yes	0.0838	0.0736
EfficientNetB0	Yes	0.0963	0.0824

B.0.3 COVID-19 Scans Dataset Tables

Table B.5: Retrieval Results for COVID-19 Scans Dataset

		KNN							FAISS						
Model	Background	Top 10 Acc.	P@5	P@10	R@5	R@10	F1@5	F1@10	Top 10 Acc.	P@5	P@10	R@5	R@10	F1@5	F1@10
ResNet50	Yes	0.97	0.36	0.36	0.87	0.97	0.49	0.51	0.96	0.36	0.36	0.87	0.96	0.49	0.51
MobileNetV2	Yes	0.96	0.36	0.36	0.86	0.96	0.48	0.50	0.96	0.36	0.36	0.87	0.96	0.49	0.50
EfficientNetB0	Yes	0.95	0.37	0.37	0.86	0.95	0.49	0.51	0.94	0.37	0.37	0.86	0.95	0.49	0.51

Table B.6: Inference Results for COVID-19 Scans Dataset

Model	Background	KNN Inference Time (s)	FAISS Inference Time (s)
ResNet50	Yes	0.1197	0.1049
MobileNetV2	Yes	0.0956	0.0726
EfficientNetB0	Yes	0.0973	0.0804

B.0.4 Pokémon Dataset Tables

Table B.7: Retrieval Results for Pokémon Dataset

		KNN							FAISS						
Model	Background	Top 10 Acc.	P@5	P@10	R@5	R@10	F1@5	F1@10	Top 10 Acc.	P@5	P@10	R@5	R@10	F1@5	F1@10
ResNet50	Yes	0.82	0.35	0.26	0.72	0.82	0.45	0.36	0.83	0.38	0.29	0.77	0.83	0.48	0.40
MobileNetV2	Yes	0.82	0.38	0.29	0.74	0.82	0.47	0.39	0.83	0.40	0.30	0.74	0.83	0.49	0.41
EfficientNetB0	Yes	0.82	0.44	0.34	0.75	0.82	0.52	0.44	0.85	0.48	0.38	0.78	0.85	0.56	0.49
ResNet50	No	0.88	0.51	0.40	0.83	0.88	0.59	0.51	0.89	0.55	0.45	0.85	0.89	0.63	0.55
MobileNetV2	No	0.85	0.46	0.36	0.80	0.85	0.55	0.46	0.86	0.48	0.38	0.81	0.86	0.56	0.48
EfficientNetB0	No	0.89	0.55	0.45	0.85	0.89	0.63	0.55	0.90	0.58	0.48	0.86	0.90	0.66	0.59

Table B.8: Inference Results for Pokémon Dataset

Model	Background	KNN Inference Time (s)	FAISS Inference Time (s)
ResNet50	Yes	0.1216	0.1068
ResNet50	No	0.1242	0.1080
MobileNetV2	Yes	0.0876	0.0755
MobileNetV2	No	0.0901	0.0761
EfficientNetB0	Yes	0.0969	0.0825
EfficientNetB0	No	0.0997	0.0840