UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
CURSO DE CIÊNCIA DA COMPUTAÇÃO

PAULO GAMARRA LESSA PINTO

# PetsRS Dataset: A Benchmark and Baseline for Pet Recognition in a Climate Disaster Scenario

Work presented in partial fulfillment of the requirements for the degree of Bachelor in Computer Science

Advisor: Prof. Dr. Claudio Rosito Jung

Porto Alegre
August 2024

## ACKNOWLEDGEMENTS

I would like to thank my parents Prof. Paulo Lessa and Prof. Blanca Gamarra for always supporting and encouraging me in pursuing higher education.

I would also like to thank my advisor Prof. Claudio Jung and my colleagues at the Computer Graphics, Image Processing, and Interaction group for guiding me in my computer vision journey.

Next, I would like to thank my girlfriend Juliana for her emotional support and for always pushing me to do better.

Lastly, I would like to thank the PetsRS team for providing me with data and great ideas.

**ABSTRACT**

In the first half of 2024, thousands of pets were separated from their guardians due to the floods in Rio Grande do Sul. In this context, the PetsRS tool emerged, which aims to help guardians in the search for their lost animals. The tool allows users (in this case, guardians or volunteers from shelters that are taking in rescued animals) to send images of the animals. Then it is possible to perform a manual or automated search, which searches its database for a "match" between a picture of a lost and rescued animal. In the present work, tuples of images of dogs and cats submitted to PetsRS are collected are cleaned, pre-processed, and organized to create an image retrieval dataset. Then, pre-trained off-the-shelf image encoders representing different training paradigms are evaluated for pet recognition using the k-NN algorithm with the embeddings as data points. In our results, DINOv2 achieves the best top-k accuracy on all tests and appears to be the best pre-trained encoder for this task. The objective of this work is to create a real benchmark and baseline for evaluating techniques that tackle pet recognition, especially the ones deployed at PetsRS.

**Keywords:** Animal identification. Image retrieval. Computer Vision. Machine Learning.

**PetsRS Dataset: Um Benchmark e Baseline para Reconhecimento de Pets em um Cenário de Desastre Climático**

## RESUMO

No primeiro semestre de 2024, milhares de animais de estimação foram separados de seus tutores por consequencia das enchentes no Rio Grande do Sul. Neste contexto, surgiu a ferramenta PetsRS, que propõe auxiliar os tutores na busca por seus animais perdidos. A ferramenta permite que os usuários (neste caso, os tutores ou voluntários de abrigos que estão acolhendo animais resgatados) enviem imagens dos animais e, então, é possível efetuar a procura manual ou automatizada, que busca em sua base de dados por um "match" entre uma imagem de animal perdido e resgatado. No presente trabalho, tuplas de imagens de cães e gatos submetidos no PetsRS são coletadas, organizadas, pré-processadas e é feita lipeza de dados para criar um *dataset* de *image retrieval*. Depois, *encodings* pré-treinados *off-the-shelf* representando diferentes paradigmas de treinamento são avaliados para reconhecimento de pets. Nos nossos resultados, DINOv2 alcançou a melhor *top-k accuracy* em todos os testes e parece ser o melhor *encoder* pré-treinado para esta tarefa O objetivo deste trabalho é criar um *benchmark* e *baseline* reais para avaliação de técnicas que atacam o reconhecimento de pets, especialmente as implementadas no PetsRS.

**Palavras-chave:** Identificação de animais. Recuperação de imagens. Visão Computacional. Aprendizado de máquina.

## LIST OF FIGURES

## LIST OF TABLES

## LIST OF ABBREVIATIONS AND ACRONYMS

| | |
|---|---|
| RS | Rio Grande do Sul |
| ML | Machine Learning |
| YOLO | You Only Look Once |
| SAM | Segment Anything Model |
| k-NN | k-nearest neighbors |
| LGPD | Lei Geral de Proteção de Dados Pessoais |
| CNN | Convolutional Neural Network |
| ViT | Vision Transformer |
| VLM | Vision-Language Model |
| MLP | Multilayer Perceptron |

**CONTENTS**

# 1 INTRODUCTION

Between the end of April and the beginning of May 2024, the state of Rio Grande do Sul (RS) in Brazil was struck by severe floods in the region surrounding the Guaíba waterbody. These floods left several cities in a state of public calamity and were classified as the most severe climate disaster in the state's history. The tragedy led to 173 deaths and more than 420 thousand people leaving their homes (Sul, 2024).

In the aftermath of the floods, local authorities and volunteers rescued thousands of pets. However, many of these pets were separated from their guardians, which led to the creation of improvised animal shelters. More than 300 pet shelters were created, and more than 18 thousand pets were rescued (Bianca, 2024). The overpopulation of these shelters presents several problems, such as diseases and violence between animals, aggravated by the decreasing number of volunteers. Several efforts arose to reunite pets with their guardians as soon as possible, such as social media profiles, group chats, and web pages where volunteers/guardians shared pictures of rescued/lost pets. This is still an issue in RS, and work on these efforts is needed.

In this context, volunteer programmers created the PetsRS application[1] (PetsRS, 2024). On one end of this application, guardians searching for their lost pets can submit pictures of the animal and fill in characteristics such as species, breed, sex, and color. On the other end, shelter volunteers and temporary guardians of rescued pets can also submit pictures and characteristics of their animals. This results in a crowd-sourced open database of lost and rescued pets that can be filtered and analyzed to find "matches" and reunite guardians with their pets. The application also uses a Machine Learning-based pet recognition solution to find similar lost and found entries based on images that could be the same individual and shows potential "matches" for each entry.

This work is part of the effort to improve the pet recognition aspect of PetsRS. This type of solution uses models to encode images into feature encodings, and quantitative evaluation is essential when choosing the most suitable model to use. However, no public dataset with images similar to this context exists. To solve this problem, this work presents a pet recognition dataset made with pictures from the PetsRS database and a baseline for encodings for this task.

[1] https://petsrs.com.br/

## 1.1 Goals

The main goal of this work is to create a dataset, evaluation pipeline, and baseline for cat and dog image encoders aimed at pet recognition using real data from a climate catastrophe scenario. This result will be helpful for PetsRS and similar applications in the RS floods context when choosing their best encoder, as well as future works on pet recognition that want to evaluate for this scenario.

To achieve this general goal, the following specific goals are defined:

1. Create a pet identification dataset by taking images grouped by individual from the PetsRS database, organizing, cleaning, and pre-processing the data.
2. Test different off-the-shelf encoders and image pre-processing methods for pet identification on the dataset to create an encoding baseline.
3. Analyze and discuss the results.

## 1.2 Chapters organization

The rest of this document is organized as follows: Chapter 2 explains the theoretical foundation of this work, defining identification tasks and introducing the models used in the experiments. Chapter 3 presents an overview of related work on pet identification datasets. Chapter 4 describes how the proposed pet recognition dataset was created. Chapter 5 describes the baseline experiments and presents the results. Chapter 6 presents an overview of this work, discussing results, limitations, and future work.

# 2 THEORETICAL FOUNDATION

In this chapter, we present the theoretical foundation for this work. We start by defining the task of instance-level recognition and the difference between other identification tasks. Then, we present the off-the-shelf encoders used on the baseline and, finally, the supplementary models used on the dataset creation.

## 2.1 Identification Tasks

The identification task done by the PetsRS website (PetsRS, 2024) and the main goal of the dataset presented here is **pet recognition**, which is an application of instance-level recognition and a type of image retrieval task. In this section, we present the description of the recognition task and the adjacent tasks of instance verification and clustering as defined in Mougeot, Li and Jia (2019).

**Verification:** In this identification task, the model is given two images and must answer whether they belong to the same instance or not. This is the type of verification done at automatic gates with face identification. For this task, a random choice algorithm would achieve 50% accuracy.

**Clustering:** In this identification task, the model receives a set of images belonging to a set of individuals and must group the images by individual. Given the embeddings for these images, this task is done by applying the k-means algorithm (Jin; Han, 2010) to the embeddings. A random choice algorithm would achieve $1/n * 100\%$ accuracy in this task, where $n$ is the number of individuals.

**Recognition** In this task, given a query image of an unknown individual and a database of candidate images belonging to known individuals, the model must retrieve the candidate images with the same identity as the query. Given the embeddings of the query and candidate images, this is an application of the k-NN algorithm (Cover; Hart, 1967). The algorithm computes the distance between the embeddings of the query image and the candidates, and then retrieves the most frequent identity of its k nearest neighbors. Alternatively, the algorithm could retrieve the k nearest neighbors' images instead of identities. In the latter option, we can display the top-k images that are most similar to the query image, which is the task of the PetsRS application (PetsRS, 2024). Hence, this option will be used for evaluation in this work.

## 2.2 Pre-trained Encoders

To perform pet recognition with the k-NN algorithm, the images must be vectorized into feature embeddings by an image encoder. In this section, we briefly introduce the off-the-shelf pre-trained encoders used in the baseline for the proposed dataset. The choice of these models is justified in Section 5.2 with descriptions of the models and training data in the context of pet recognition. The encoders are briefly described next.

**ResNet:** A family of image classification convolutional neural networks (CNN) introduced in He et al. (2016) and trained in a fully supervised manner. The main novelty of ResNet models is the introduction of residual connections, where the input of a convolution block is also added to its output, propagating the original signal. This was a successful attempt of attacking the vanishing gradient problem that prevented the training of deeper CNN's. For this reason, ResNet models can have very deep architecture such as ResNet-152 with 152 layers. ResNet models pre-trained on ImageNet (Deng et al., 2009) have been used as backbone for a variety of tasks (Goldblum et al., 2023), including pet recognition in Mougeot, Li and Jia (2019).

**CLIP** (Radford et al., 2021): CLIP is a family of image and text encoders developed by OpenAI that are trained using contrastive learning to create image and text embeddings in a shared vector space. The training is done with image and caption pairs scraped from the Internet that the multi-modal scheme has to encode into a shared space, so that corresponding image-text pair representations have close positions. This is done via contrastive loss that encourages the cosine similarity between the correct image-text pairs to be high while reducing the similarity for incorrect pairs. The results is a set of encoders with a powerful semantic representation of images that can be leveraged for several downstream tasks in a zero-shot manner (Goldblum et al., 2023).

**DINOv2** (Oquab et al., 2024) : is a family of self-supervised ViT continuing the DINO family (Caron et al., 2021). Dino is trained to create powerful image embeddings without any labels. Like CLIP, these embeddings are very generalizable and can be used for several downstream tasks (Caron et al., 2021) (Oquab et al., 2024). The training is done via the teacher-student framework and extreme data augmentation. In this framework, two models, the student, and the teacher are trained simultaneously to generate embeddings. The teacher generates embeddings for the student to learn from. The teacher model's parameters are updated more slowly using a momentum mechanism, which allows it to produce consistent outputs across different views of the same image, even under

strong augmentations. The student model is trained to match the teacher's outputs, and as training progresses, the student gradually learns to produce high-quality representations. In the end, the student is the final model.

## 2.3 Other Relevant Models

Detecting and segmenting pets are important pre-processing tasks in creating our proposed dataset, as will be shown in Sections 4.3 and 4.2. In this section, we briefly introduce the two models used for these tasks.

**YOLO** (Redmon et al., 2016): is a family of fast object detectors and in the most recent iterations also performing instance segmentation (Terven; Córdova-Esparza; Romero-González, 2023) (Wang et al., 2024). Their main characteristic is that of being one-stage detectors, treating the detection problem as a regression one and generating bounding boxes and class probabilities in a single forward pass. This differentiates them from the traditional two-stage detector approach of region-proposals that are later classified like R-CNN (Girshick et al., 2014). However, YOLO models do not sacrifice accuracy to achieve this, making this type of model very popular for several industry applications (Terven; Córdova-Esparza; Romero-González, 2023).

**SAM** (Kirillov et al., 2023): Segment anything model (SAM) is a general purpose and promptable segmentation model. The main advantages of SAM rely on its generalization capacity since it is not limited by any task-specific segmentation training and its ability to take prompts in the form of points, bounding boxes, or text that guide the desired segmentation. SAM is composed of a ViT image encoder that encodes the image embeddings, an MLP that encodes the prompts and a transformer decoder that generates the segmentation mask according to the image and prompt embeddings.

# 3 RELATED WORK

The work on pet identification is very limited and most available pet datasets focus on pets as groups, such as breed classes, and not as individuals (Moreira et al., 2017; Mougeot; Li; Jia, 2019). In this chapter, we present an overview of available datasets for pet identification.

Moreira et al. (2017) introduced one of the earliest works on dog identification, and they present two small recognition datasets: Flickr-dog and Snoopybook. Flickr-dog is composed of 374 photos belonging to 42 individuals. Half of the individuals are of the Husky breed and the other of the Pug breed. They crop the faces to remove background information that could give unfair clues to classifiers. Snoopybook is composed of 251 pictures belonging to 18 mongrel dogs and is complementary to Flickr-dog as it offers a less controlled array of individuals. Moreira et al. (2017) also experiments with human face recognition techniques on their datasets and compares them to their two proposed encoders, a trained SVM and an off-the-shelf CNN.

Azizi and Zaman (2023) proposed a dataset for pet recognition with both dogs and cats. The dataset is an expansion of Flickr-dog with the addition of 1100 images of 70 pets that were crowd-sourced, 1800 images of 57 pets that were taken from animal shelter websites, and 521 pictures of 76 animals were taken from the internet. There is no information on the number of individuals belonging to the dog or cat species and this dataset is not publicly available. They then propose a pipeline of several techniques for classification and identification and evaluate it on their dataset.

Regarding dataset availability, DogFaceNet (Mougeot; Li; Jia, 2019) is the most relevant work to tackle the problem of animal face recognition. This paper presents a dataset of 3,148 dog face images corresponding to 485 individual dogs. After the original publication, this dataset has been increased, and nowadays it is composed of a total of 8,363 face images corresponding to 1,393 dogs. These images were all taken by the authors or collected from pet adoption websites. Moreover, they train a ResNet-inspired (He et al., 2016) encoder using the triplet loss (Schroff; Kalenichenko; Philbin, 2015) and evaluate it for dog face verification, recognition, and clustering. Yoon, So and Rhee (2021) expanded the DogFaceNet encoders by proposing a new vector space that is not limited to a unit sphere as is the case for normalized vectors.

Some works use nose-print datasets for both dog (Bae; Pak; Lee, 2021; Caya; Arturo; Bautista, 2021) and cat (Chen et al., 2016) identification. The nose-print identifies

the animals in the same manner that fingerprints identify humans. Bae, Pak and Lee (2021) presents a dataset of 2,561 dog nose-print images from 302 individuals and Chen et al. (2016) presents a dataset of 700 nose-prints belonging to 70 individual cats.

As presented in this chapter, there is a lack of available identification datasets for pets (especially cats), with DogFaceNet being the only large example. Moreover, in the context of the PetsRS project, whole-body data and encoding techniques are needed, since there is no guarantee that the faces or nose-prints are fully visible and aligned in all submissions. This shows the competitiveness and importance of the dataset proposed in this work, which is the only large dataset of both dog and cat whole-body images.

# 4 DATASET

This chapter describes the process of creating our image-based pet recognition benchmark dataset. The process begins by downloading raw data from the PetsRS web page. Then, this data is cleaned to remove problematic samples (that are explained and exemplified in Section 4.2) and pre-processed by cropping a region of interest defined by a pet bounding box and removing the background. Finally, the cleaned data are organized in "query" and "candidate" pairs for image retrieval using pet recognition evaluation.

In the PetsRS data, there is no information about potential "matches" between lost and found dogs, i.e., individuals who are both on the lost and found sets. Hence, creating a dataset where one "query" image is a real lost dog and its corresponding correct "candidate" is a real found dog is impossible. The proposed solution is to create two separate datasets, one with only the pets submitted as lost and another with the ones submitted as lost. Pets can be submitted with more than one image in the PetsRS website, and the data from these pets is used to create corresponding image pairs representing the same individual. This separation also guarantees that possible correct pairings between two images of the same individual submitted as lost and found are not computed as incorrect during evaluation. The steps used in the dataset creation are detailed next.

## 4.1 PetsRS data

The images were downloaded directly from the PetsRS web page (PetsRS, 2024) employing a Bash script provided to registered volunteers on their private repository. The script downloads all images of all individuals in the structure shown in Figure 4.1. The first-level two folders denote whether an individual was submitted as a lost pet currently being searched for by its guardians or as a found pet that has been rescued and is currently in either a shelter or a temporary home. Then, both folders are divided into sub-folders denoting the species of the individuals, which could be either "dog," "cat," "horse," "bird," or "other." The images are inside these folders with names following the format "x.png" or "x_y.png", where x is the pet ID and y is the image ID (which is omitted when the ID is 1).

Information about the amount of images in the raw data is presented in Table 4.1. The table shows that most of the images are of dogs, followed by cats, and then horses, birds, and other animals. The last three have too few samples for an instance-level

```
PetsRS Data/
├── lost/
│   ├── dog/
│   │   ├── 1.png
│   │   ├── 1_2.png
│   │   ├── 1_3.png
│   │   ├── 2.png
│   │   ├── 2_2.png
│   │   └── ...
│   └── cat/
│       └── ...
└── found/
    ├── dog/
    │   └── ...
    └── cat/
        └── ...
```

Figure 4.1 – Folder structure of the raw PetsRS data.

recognition dataset.

As explained at the beginning of this chapter, our interest is in pets with more than one image, as these are the ones that could be used to create image retrieval pairs. Also, species other than dogs and cats have too few samples to compose a relevant dataset. We discard pets with only a single image or of a class other than cat and dog.Information about the number of images that remained after this step is presented in Table 4.2. The table shows that a significant number of images were deleted when compared to Table 4.1. However, the dog and cat classes still have a significant amount of individuals when compared to other pet datasets (Mougeot; Li; Jia, 2019).

A challenge to this strategy is present in pets with all images on similar backgrounds since an encoder could use the background features embeddings and correctly pair images without the individual's identity embeddings. This problem is especially present on the "found" pets since these pictures were taken in shelters, usually in one session, so there is not a lot of scene variety, as shown in examples in Figure 4.3. This is not so common on the "lost" pets since these are pictures taken by their guardians over extended periods.

Table 4.1 – Statistics for the data downloaded from PetsRS. Table entries are in the format $n/m$ where $n$ is the number of individuals and $m$ is the number of images.

| | total | found | lost |
|---|---|---|---|
| dog | 6577 / 10721 | 4516 / 7426 | 2061 / 3295 |
| cat | 1757 / 3039 | 480 / 776 | 1277 / 2263 |
| horse | 16 / 25 | 3 / 3 | 13 / 22 |
| bird | 14 / 24 | 1 / 1 | 13 / 23 |
| other | 4 / 5 | 3 / 3 | 1 / 2 |
| total | 8368 / 13814 | 5003 / 8209 | 3365 / 5605 |

Source: The Author

Table 4.2 – Statistics for the data from pets with more than one image on their entry. Table entries are in the format $n/m$ where $n$ is the number of individuals and $m$ is the number of images.

| | total | found | lost | mean #images | median #images |
|---|---|---|---|---|---|
| dog | 1982 / 6126 | 1280 / 4190 | 702 / 1936 | 3.0908 | 3.0 |
| cat | 669 / 1951 | 181 / 477 | 488 / 1474 | 2.9162 | 3.0 |
| total | 2664 / 8110 | 1461 / 4667 | 1203 / 3443 | 3.0442 | 3.0 |

Source: The Author

## 4.2 Data cleaning

In this section, we describe the steps taken to clean the raw data downloaded from PetsRS to create the set of images used in the final dataset. The main problems present in the raw data are listed below and will be further explained in the next subsections.

- Corrupted files
- Duplicates and near duplicates
- Images without pets or with more than one pet

### 4.2.1 Rewriting data

The Bash script that downloads the PetsRS data writes all images with the suffix ".png". However, not all images are in the PNG format, which leads to problems when opening these images with software such as Ubuntu's image viewer. Moreover, some images were corrupted. Python's `OpenCV` package can open these images with the wrong suffix, so to fix this issue, all images were rewritten as PNG files using OpenCV. Also, corrupted images that OpenCV could not read were deleted.

Figure 4.2 – Example of a near-duplicate pair of pet images.

### 4.2.2 Removing duplicates and near-duplicates

A quick exploration of the PetsRS data revealed two types of duplicates. First, some users submitted pets more than once, which is problematic for the proposed dataset because correct pairings of images from the same individual would be computed as incorrect. Second, other pets were submitted with duplicate or near-duplicate images, which is problematic because a correct pairing would be too easy. In the context of these data, near-duplicates are pictures that seem to have been taken almost simultaneously with very little movement between each shot, as exemplified in Figure 4.2. These pairs are not a good representation of real lost and found entries of an individual because it is unlikely that all conditions such as lighting, background, shot angle and pet pose would be so similar.

The `findimagedupes` command line application was used to remove these duplicates (Chin, 2006). This application initially computes a *fingerprint* for each image using the following steps:

1. Read image.
2. Resample to $160 \times 160$ to standardize size.
3. Grayscale by reducing saturation.
4. Blur a lot to get rid of noise.
5. Normalize to spread out intensity as much as possible.
6. Equalize to make the image as contrasty as possible.
7. Resample again down to $16 \times 16$.
8. Reduce to 1bpp.
9. The fingerprint is this raw image data.

Then, it detects both duplicates and near duplicates by comparing the fingerprints of each pair of images. The comparison is achieved through the following steps:

1. Take fingerprint pairs and `xor` them.
2. Compute the percentage $p$ of bits zero in the result.
3. If $p > T_p$, declare files to be similar, where $T_p$ is a threshold (in this work, $T_p$ was set to its default value of 90%).

The result is a text file with all the duplicates and near-duplicate image tuples. These detections were first human-reviewed to avoid deleting wrong duplicate detections and as a data exploration mechanism; however, because of time limitations, this review was interrupted. Manual review of the detections revealed a very small number of cases where `findimagedupes` generated false positives, i.e., tuples that are accused of being duplicates but are not. Therefore, automatic processing of duplicate detections would not remove a significant amount of samples wrongly detected as duplicates. The duplicate detections for pets submitted as lost were human-reviewed, while a Python script processed the ones for pets submitted as found. The Python script followed the steps below for each image tuple detected as duplicate:

1. If all images belong to the same individual, keep one and go to step 2. Else, go to step 3.
2. If the individual ended up with only one image, remove the individual from the dataset and end the script.
3. If duplicates belong to different individuals, keep the individual with more images and remove others. End script.

Table 4.3 shows the number of images after removing duplicates and pets that ended up with only one image. Again, the number of instances is still competitive when compared to the number of images in related datasets. However, the median number of images has gone from 3 to 2, which means that the dataset is now more suited for one-shot recognition

### 4.2.3 Removing images with no pets or more than one pet

Some images submitted to the PetsRS website do not present any animal. Most of these are images of textual information with contact information of the guardian searching

Table 4.3 – Statistics for the data after removing duplicates and individuals who ended up with only one image after duplicates removal. Table entries are in the format $n/m$ where $n$ is the number of individuals and $m$ is the number of images.

| | total | found | lost | mean #images | median #images |
|---|---|---|---|---|---|
| dog | 1857 / 5651 | 1207 / 3889 | 650 / 1762 | 3.0430 | 2.0 |
| cat | 618 / 1802 | 172 / 456 | 446 / 1346 | 2.9158 | 2.0 |
| total | 2475 / 7453 | 1379 / 4345 | 1096 / 3108 | 3.0113 | 2.0 |

Source: The Author

for the pet or the shelter where the pet is being rescued. Other images may present more than one pet, such as an animal in the foreground and others in the background, or more than one animal on the same plane.

To solve these issues, we explore a pre-trained object detector with datasets that contain animal-related categories. More precisely, we use YOLOv8 (You Only Look Once) (Terven; Córdova-Esparza; Romero-González, 2023) to detect bounding boxes belonging to either the "dog", "cat", "horse", or "teddy bear" classes. Any image where none of these classes was detected was removed from the dataset. "horse" and "teddy bear" were added as interest classes because, in the experiments, several dogs were classified as these classes. A model from the YOLO family (Redmon et al., 2016) developed by Ultralytics was used here because of their ease of use, generalization capacity, and fast execution (which would leave more time for segmentation on Section 4.3) (Zaidi et al., 2022; Terven; Córdova-Esparza; Romero-González, 2023). YOLOv10 (Wang et al., 2024) was also experimented but it did not detect as many images with pets as YOLOv8, and we prioritized keeping images.

For each image, the bounding box estimated by YOLOv8 with the highest confidence was used to crop the images to remove background information and other individuals, keeping only the main target pet. This technique does not guarantee the correction of cases where more than one pet is in the same plane. For instance, if two images with two pets each are submitted as one individual, there is no guarantee that the highest-confidence box in both images relate to the same pet. This would result in a pair of images wrongly identified as the same individual.

A possible solution is also to delete images where YOLO detects more than one pet bounding box; however this would remove images with pets on the background who could be cropped out. Human-assisted review should be implemented to solve this issue, but this problem can be present in this dataset's version.

This cleaning step was performed simultaneously with creating the two datasets with different pre-processing techniques described in Section 4.3. Table 4.4 shows the

Table 4.4 – Statistics for the data after removing pictures where YOLO did not detect a pet and the ones who ended up with only 1 image after this removal. Table entries are in the format $n/m$ where $n$ is the number of individuals and $m$ is the number of images.

| | total | found | lost | mean #images | median #images |
|---|---|---|---|---|---|
| dog | 1834 / 5546 | 1201 / 3838 | 633 / 1708 | 3.0239 | 2.0 |
| cat | 616 / 1789 | 172 / 454 | 444 / 1335 | 2.9042 | 2.0 |
| total | 2450 / 7335 | 1373 / 4292 | 1077 / 3043 | 2.9939 | 2.0 |

Source: The Author

number of images after removing the ones where YOLOv8 detected no pets, which keeps being competitive. The number of individuals presented in this table is the final number of individuals for this dataset. Compared to Table 4.1, we can see that data cleaning removed 6361 images belonging to 5918 individuals.

### 4.3 Background removal

Images in the PetsRS dataset have elements and information unrelated to the pet, such as background information or other individuals (pet or human). This background information could be a problem for the encoder that creates the image embeddings because it is not part of the individual's identity. Moreover, some pictures of the pets are all in the same environment, as shown on Figure 4.3, so an encoder could be influenced by the background to achieve a correct retrieval without considering the pet's identity.

After cleaning the PetsRS data, we generated two versions of the dataset with different image pre-processing steps to mitigate these issues at different levels. These versions are the **crop-original** and **crop-segmented** sets.

The crop-original set was created by taking the cleaned images and cropping the bounding box with the highest confidence, as described in Section 4.2.3. This guarantees that only one individual is framed on each image and removes *some* unnecessary background information. However, just a crop still retains some background information. The crop-segmented set was created to mitigate these issues and provide a fairer and more challenging evaluation where only the pet identity information is present in the image. It is created by coupling YOLO v8 with SAM (Segment Anything Model) (Kirillov et al., 2023) to segment the pet and remove the background. The SAM "vit_h" model was used, which is the one with the highest segmentation accuracy and slowest execution time since the priority is for a good segmentation result. This model can receive a bounding box as guidance, which is refined into a more precise segmentation mask. The output segmenta-

tion mask from SAM is then used as a pixel-wise binary mask to remove the background from each pet image. Some examples of YOLO's bounding box, SAM's segmentation mask, and resulting crop-original and crop-segmented data are shown in Figure 4.4.

## 4.4 Dataset organization

After data cleaning and background processing, the resulting dataset is composed of two sets of images depending on pre-processing (crop-original and crop-segmented), which are divided into two subsets depending on whether the individual is lost or found and also divided into subsets for each species (cat and dog). To create the final pet-recognition-based image retrieval dataset, the two images with the lowest ID for each individual are selected to compose a query/candidate pair. We chose to select the two images with the lowest ID because PetsRS volunteers were instructed to submit the images where the pet is best framed as the first ones, with the one with the lowest ID being the "main" image. The final dataset is a collection of eight different isolated and self-contained datasets with different types of pre-processing (with or without background), origin (lost or found) and species (cat or dog). The folder structure of this dataset is presented in Figure 4.5, showing the eight different query/candidate datasets and the file format **pet_id**.png. The number of query/candidate pairs is presented in Table 4.5.

Table 4.5 – Stats for the final dataset. Each table entry is the number of query/candidate pairs for each subset of the dataset.

| | total | found | lost |
|---|---|---|---|
| dog | 1834 | 1201 | 633 |
| cat | 616 | 172 | 444 |
| total | 2450 | 1373 | 1077 |

Source: The Author

## 4.5 LGPD

*Lei Geral de Proteção de Dados Pessoais (LGPD)* (Brazil, 2018) is the Brazilian legislation regarding the personal data of individuals in the country. It establishes the rules for collecting and sharing it. These rules do not apply to animals, so we could make the dataset presented in this work available without worrying about privacy complications. However, some images contain human faces that may not have been deleted by the steps

described in Section 4.3. Assuring these faces are not in the dataset would require a time-consuming human review of all images. For this reason, the dataset is not publicly available at the time of this publication, but we intend to do it in the future.

Figure 4.3 – Examples of pets with images on the same background. Each row of images belongs to an individual pet.

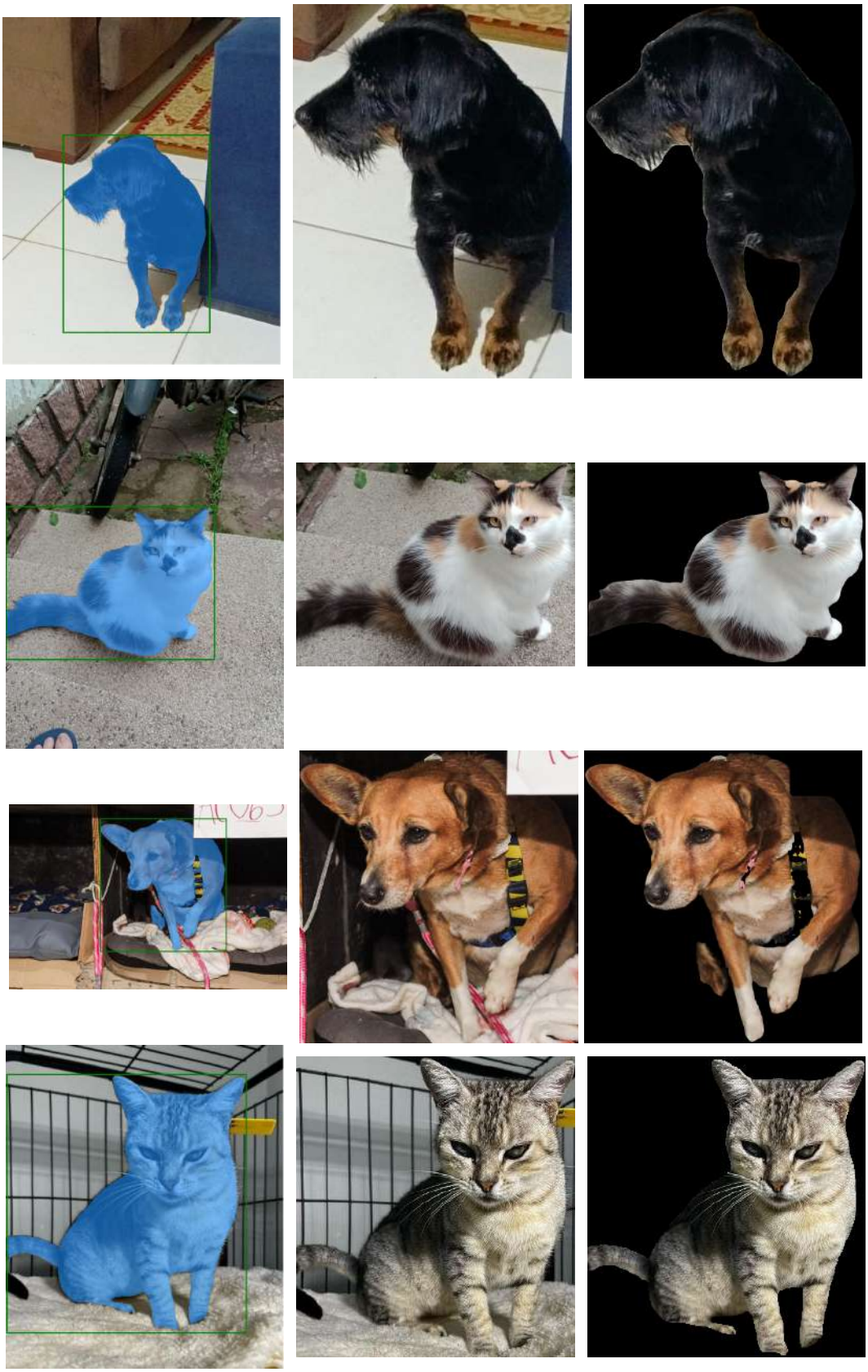

Figure 4.4 – Examples of the background removal process. For each row, the left most image shows YOLO's bounding box in green and SAM's segmentation mask in blue, the middle image shows the pet crop and the right most image shows the pet crop and background removal.

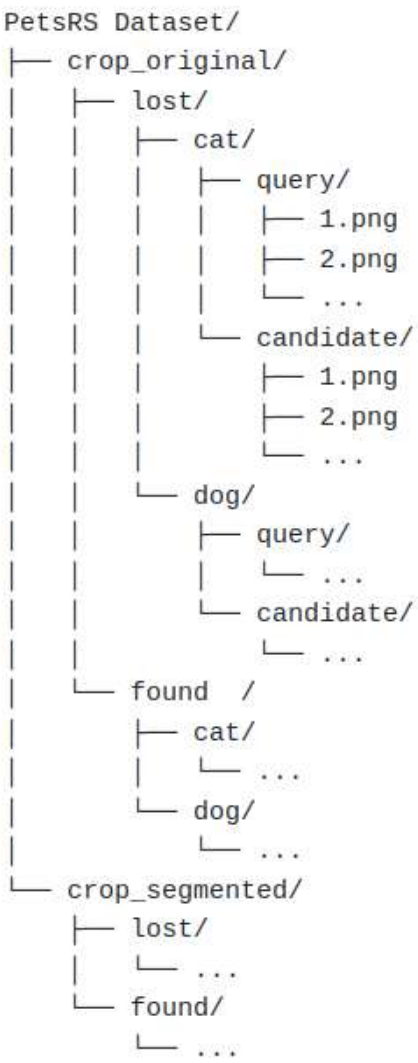


Figure 4.5 – Folder structure of the proposed PetsRS dataset.

# 5 EXPERIMENTS

In this chapter, we evaluate several pre-trained image encoders to create a baseline on our dataset that can be used for future work comparisons. We explain and justify the evaluation on both the **crop-original** and **crop-segmented** in section 5.1. In Section 5.2, we list and justify the choice of encoders for these experiments. We follow the image recognition evaluation process presented in Mougeot, Li and Jia (2019), which uses the k-NN (Cover; Hart, 1967) algorithm and evaluates with rank-k accuracy; however, we only evaluate one-shot recognition (i.e., only one image per pet is used to train the k-NN algorithm) since our dataset is composed of pairs of images.

## 5.1 The Impact of Pre-processing

As described in Section 4.3, all images have two versions belonging to datasets **crop-original** and **crop-segmented**, respectively. These two versions of the dataset could be used to check what level of pre-processing would achieve the best results when coupled with each encoder. However, as exemplified in Figure 4.3, some individuals have their images taken with the same background.

Encoders could make correct matches using the background features instead of the individuals' identities when using images with backgrounds. Because of this, the crop-segmented dataset represents a fairer evaluation of an encoder's identification capacity. Experiments were done with both the crop-segmented and crop-original datasets, and the effect of each on accuracy is explored on Section 5.4

## 5.2 Pre-trained Encoders

Our experiments used three pre-trained models as encoders: Resnet (He et al., 2016), DINOv2 (Oquab et al., 2024), and CLIP (Radford et al., 2021). These models represent different training paradigms, datasets, and neural network architectures, forming a lean but comprehensive benchmark of available off-the-shelf general-purpose image encoders (Goldblum et al., 2023).

**ResNet** is a family of convolutional neural network (CNN) (LeCun et al., 1989) image classification models trained in a supervised manner on the ImageNet dataset (Deng

et al., 2009) with 1,000 classes. ResNet and ResNet-like models pre-trained on ImageNet have been commonly used as backbones for several computer vision tasks (Goldblum et al., 2023), including pet recognition Mougeot, Li and Jia (2019). Hundreds of dog breeds, a few cat breeds, and hundreds of other animals are present in the ImageNet classes. Therefore, it is justified that feature embeddings from a ResNet model could capture a pet's identity. We used ResNet-152, the architecture with the best classification accuracy when evaluated on ImageNet(He et al., 2016), and took the output of its penultimate layer as embedding.

**CLIP**: CLIP is a family of multi-modal models trained with contrastive multi-modal learning to align image and text pairs in a shared feature space. As with DinoV2, the training strategy of CLIP makes it very generalizable to different tasks, as it is not limited to task-specific labels. CLIP embeddings have been evaluated with good results for Text-Image retrieval and zero-shot image classification, which, coupled with its generalization, indicates potential for pet recognition. We used CLIP ViT-L-14@336px, which is their best overall model (Radford et al., 2021).

**DINOv2** is a family of representation learning vision transformers (ViT) (Dosovitskiy et al., 2021) trained with self-supervision on a proprietary dataset, having the potential to capture richer feature embeddings that are not limited by task-specific labels (Caron et al., 2021; Oquab et al., 2024). DINO models have been evaluated and perform well for instance-level recognition but have not been evaluated explicitly for pet recognition. We used DINOv2 ViT-L/14, the architecture that achieved the best mAP for instance-level recognition on most assessed datasets in Oquab et al. (2024).

### 5.3 k-NN evaluation

For each of the eight datasets, we create feature embeddings for the query and candidate images using each pre-trained encoder. ResNet, CLIP, and DINOv2 have feature embeddings of size 2048, 768, and 1024, respectively. Then, a k-NN algorithm is trained using the candidate images for each resulting set of embeddings. Finally, for each query embedding, the k nearest neighbors are computed and used for rank-k accuracy.

In the experiments, we used $k = 1$ and $k = 5$ as in Mougeot, Li and Jia (2019) and $k = 20$, more representative of the PetsRS scenario. In this scenario, a guardian searching for their lost pet would not take much time to check the 20 most similar found pets, so having a good rank-20 accuracy would be good enough for the application.

As for the distance metric, the cosine distance was used for DINOv2 and CLIP as their embeddings are normalized during training; therefore, the vector magnitude is irrelevant (Oquab et al., 2024; Radford et al., 2021). For ResNet, however, both cosine and Euclidean distance were tested because the high-dimensional feature vector used as embedding is not normalized, and each metric would capture different aspects of this encoding.

## 5.4 Results

Table 5.1 – Rank-1 accuracy for all encoders across all datasets (%)

| encoder | lost | | | | found | | | |
|---|---|---|---|---|---|---|---|---|
| | original | | segmented | | original | | segmented | |
| | dog | cat | dog | cat | dog | cat | dog | cat |
| ResNet-Cosine | 40.44 | 26.13 | 33.97 | 20.95 | 53.54 | 41.86 | 43.46 | 33.14 |
| ResNet-Euclidean | 36.49 | 25.45 | 32.54 | 19.14 | 50.46 | 41.86 | 41.88 | 31.98 |
| CLIP | 33.33 | 25.90 | 29.54 | 23.65 | 53.62 | 43.02 | 39.80 | 39.53 |
| DINOv2 | **52.92** | **38.06** | **48.66** | **35.59** | **68.28** | **58.14** | **61.20** | **51.16** |

Table 5.2 – Rank-5 accuracy for all encoders across all datasets (%)

| encoder | lost | | | | found | | | |
|---|---|---|---|---|---|---|---|---|
| | original | | segmented | | original | | segmented | |
| | dog | cat | dog | cat | dog | cat | dog | cat |
| ResNet-Cosine | 60.98 | 39.19 | 50.71 | 31.08 | 70.52 | 58.72 | 60.45 | 51.74 |
| ResNet-Euclidean | 55.92 | 36.04 | 48.34 | 29.73 | 66.61 | 54.07 | 58.70 | 50.58 |
| CLIP | 46.13 | 38.96 | 43.13 | 37.16 | 65.45 | 58.72 | 52.37 | 54.65 |
| DINOv2 | **72.67** | **63.06** | **68.56** | **57.21** | **81.77** | **81.98** | **76.85** | **72.67** |

Table 5.3 – Rank-20 accuracy for all encoders across all datasets (%)

| encoder | lost | | | | found | | | |
|---|---|---|---|---|---|---|---|---|
| | original | | segmented | | original | | segmented | |
| | dog | cat | dog | cat | dog | cat | dog | cat |
| ResNet-Cosine | 78.83 | 54.50 | 67.93 | 48.42 | 82.10 | 73.84 | 74.69 | 63.37 |
| ResNet-Euclidean | 74.09 | 50.90 | 63.82 | 45.05 | 78.60 | 66.28 | 71.02 | 60.47 |
| CLIP | 59.40 | 55.63 | 56.40 | 51.80 | 74.02 | 78.49 | 63.03 | 65.12 |
| DINOv2 | **86.73** | **82.66** | **84.20** | **79.28** | **89.43** | **95.93** | **84.85** | **90.12** |

Tables 5.1, 5.2, 5.3 present the rank-k accuracy for each encoder across all datasets using $k = 1$, $k = 5$ and $k = 20$, respectively. The results show that DINOv2 presents the best embeddings for pet recognition on this dataset among the models evaluated. For all datasets and k values, DINOv2 obtained the best accuracy by a significant margin.

When the results of the crop-original and crop-segmented datasets are compared, it is noticeable that all models are less accurate on the crop-original datasets. This could be because these models were not trained on images without backgrounds and, therefore, perform worse with them removed, or because of pets that have pictures with very similar backgrounds that are easier to be correctly paired, as shown in Figure 4.3.

As for the different origins (lost or found), all models have higher accuracy on the found datasets. This phenomenon can be explained by the fact that found pets tend to have more images in the same environment and background, which, therefore, represents a more manageable task as the model can also take the features of the environment into account for the embeddings. To counter this hypothesis, there is the fact that even crop-segmented results are better for found pets , which indicates that the found dataset might be essentially easier or segmentation is still leaving some background information. A possible explanation is that pets are in similar positions or wearing the same clothes in several of the found pictures because they were taken at the same moment.

When comparing the results between species, we can see that CLIP is the model with the most similar results between cats and dogs across all datasets. ResNet and DINOv2 mostly have better rank-1 and rank-5 results for dogs and get consistently similar results or better cat results only at rank-20; however, at rank-20, each set of cat nearest neighbors represent 11.6% and 4.5% of the candidates for found and lost respectively, making it a much easier task. The ResNet performance discrepancy between cats and dogs can be explained by the fact that ImageNet has more than a hundred breeds of dogs as classes while only having five cat breed classes.

After this analysis, the metrics on the "lost-segmented" column seem to be the fairest (in the sense of better representing a real-world scenario) when it comes to evaluating an encoder's capacity for pet identification, and DINOv2 is the most suited off-the-shelf encoding for the task among the models tested. Figure 5.1 shows some top-5 candidates for some queries using DINOv2 with segmented data, where the query images are in blue frames and their respective correct retrieval candidate is in a red frame if it is in the top 5 results. In the quantitative examples, we can see that the embeddings retrieve correct pairings even for instances where both images have different backgrounds and the animals are in different poses. For the wrong retrievals, the embeddings still retrieve animals with similar characteristics, such as fur patterns.

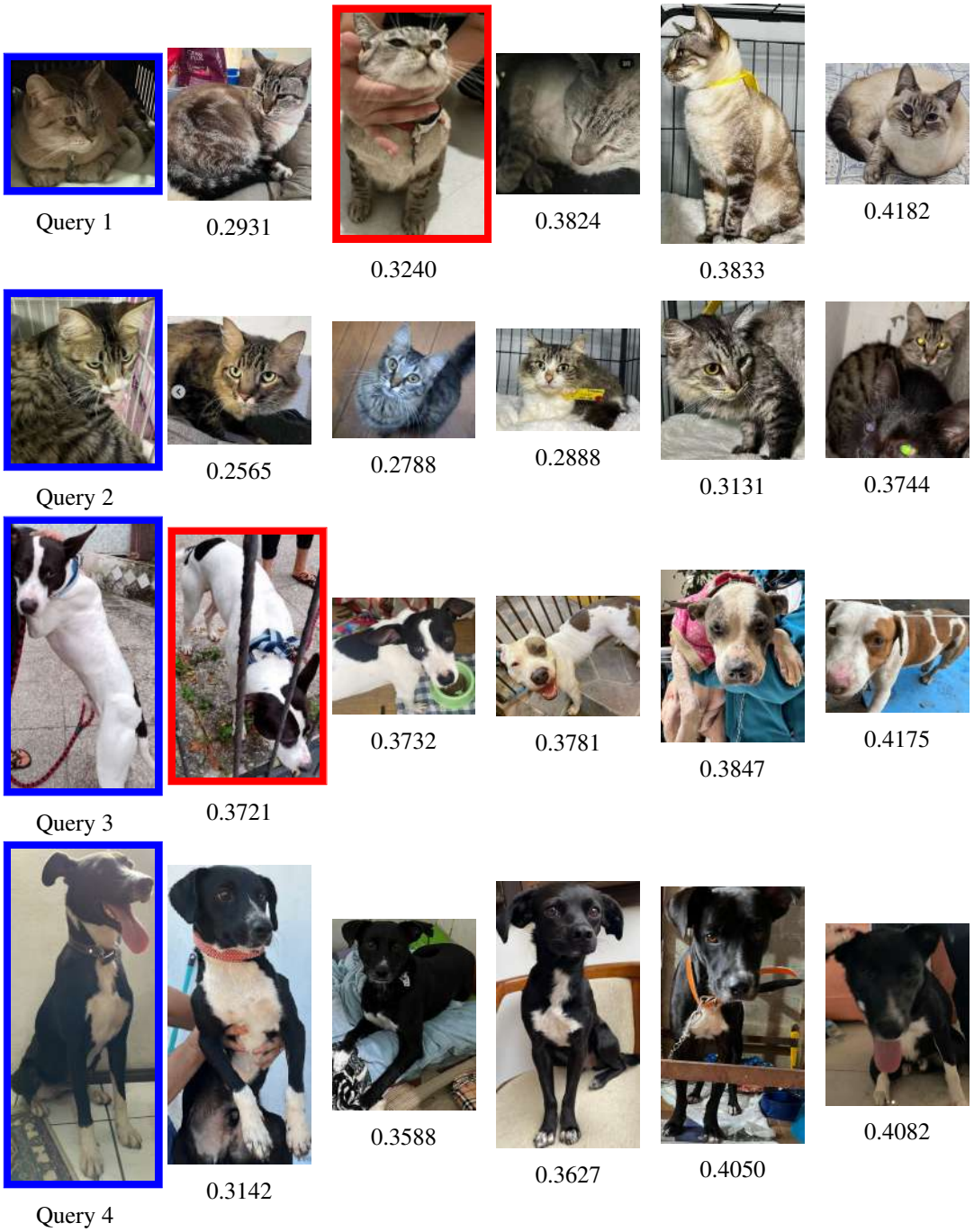


Figure 5.1 – Examples of query images on a blue frame and their top-5 results with correct retrievals on a red frame if found. The cosine distance between each candidate and the query is shown under each corresponding candidate.

Results were generated using DINOv2 embeddings and the segmented input.

# 6 CONCLUSIONS AND FUTURE WORK

In this work, we presented a pet instance-level recognition dataset with a competitive amount of instances and images when compared to public pet recognition datasets (Mougeot; Li; Jia, 2019). The dataset is composed of images taken from PetsRS, a recognition-based image retrieval application developed in the context of the 2024 RS floods. These images represent data from a catastrophe scenario and are a useful benchmark for models intended to be used in similar situations. The dataset can be directly used by PetsRS developers to evaluate their embeddings and will be published publically after manual review to remove human faces.

To create a pet recognition baseline, we evaluated three different encoders with the k-NN algorithm, each representing distinct neural network architectures and training paradigms: ResNet, CLIP, and DINOv2. The results were analyzed and discussed to show the strengths and weaknesses of each model, with DINOv2 achieving the best results for the off-the-shelf models evaluated.

In future work, we intend to organize the dataset in pre-defined train and validation splits to test training identification embeddings on it and compare the model to the baseline presented here. Moreover, manual cleaning of the dataset must be done to eliminate human faces and enable the publication of the data without legal concerns regarding LGPD. Furthermore, findimagedupes needs to be applied to the dataset again because some duplicates were created after the cropping stage. Some pets had images that were collages containing other of its images, and, after cropping the pet bounding box, the collage process was reverted, resulting in new duplicates.

As for expanding the baseline, a pet face detection algorithm can be used to evaluate pet face recognition models such as (Mougeot; Li; Jia, 2019). Additionally, it would be interesting to evaluate the impact of test time augmentation (Kimura, 2021).

As an ablation study, it would be interesting to create a dataset that is the inverse of crop-segmented, where the pet is removed from the picture leaving only the background. Evaluating models on this dataset would show how much the background can influence unfair correct retrievals of instances that had both query and candidate pictures taken on the same background.

Finally, to visualize what influences a correct retrieval, techniques such as Grad-CAM (Selvaraju et al., 2017) for CNN's and attention maps (Dosovitskiy et al., 2021) for ViT's should be applied. This would provide insight into both problems on the dataset

and what type of architectures have the potential for improvement with better data.

# REFERENCES

AZIZI, E.; ZAMAN, L. Deep learning pet identification using face and body. **Information**, v. 14, n. 5, 2023. ISSN 2078-2489. Available from Internet: <https://www.mdpi.com/2078-2489/14/5/278>.

BAE, H. B.; PAK, D.; LEE, S. Dog nose-print identification using deep neural networks. **IEEE Access**, v. 9, p. 49141–49153, 2021.

BIANCA, D. **RS ainda tem 18,4 mil animais em abrigos; para entidade de proteção, situação é "crítica e delicada"**. 2024. Available from Internet: <https://gauchazh.clicrbs.com.br/ambiente/noticia/2024/07/rs-ainda-tem-184-mil-animais-em-abrigos-para-entidade-de-protecao-situacao-e-critica-e-delicada-clygc5
html>.

BRAZIL. **Lei Geral de Proteção de Dados Pessoais (LGPD)**. 2018. Available from Internet: <https://www.planalto.gov.br/ccivil_03/_ato2015-2018/2018/lei/l13709.htm>.

CARON, M. et al. Emerging properties in self-supervised vision transformers. In: **2021 IEEE/CVF International Conference on Computer Vision (ICCV)**. [S.l.: s.n.], 2021. p. 9630–9640.

CAYA, M. V.; ARTURO, E. D.; BAUTISTA, C. Q. Dog identification system using nose print biometrics. In: **2021 IEEE 13th International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment, and Management (HNICEM)**. [S.l.: s.n.], 2021. p. 1–6.

CHEN, Y.-C. et al. Locality constrained sparse representation for cat recognition. In: TIAN, Q. et al. (Ed.). **MultiMedia Modeling**. Cham: Springer International Publishing, 2016. p. 140–151. ISBN 978-3-319-27674-8.

CHIN, J. H. N. **findimagedupes - Finds visually similar or duplicate images**. 2006. Available from Internet: <https://github.com/jhnc/findimagedupes>.

COVER, T.; HART, P. Nearest neighbor pattern classification. **IEEE Transactions on Information Theory**, v. 13, n. 1, p. 21–27, 1967.

DENG, J. et al. Imagenet: A large-scale hierarchical image database. In: **2009 IEEE Conference on Computer Vision and Pattern Recognition**. [S.l.: s.n.], 2009. p. 248–255.

DOSOVITSKIY, A. et al. **An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale**. 2021. Available from Internet: <https://arxiv.org/abs/2010.11929>.

GIRSHICK, R. et al. Rich feature hierarchies for accurate object detection and semantic segmentation. In: **2014 IEEE Conference on Computer Vision and Pattern Recognition**. [S.l.: s.n.], 2014. p. 580–587.

GOLDBLUM, M. et al. Battle of the backbones: A large-scale comparison of pre-trained models across computer vision tasks. In: OH, A. et al. (Ed.). **Advances in Neural Information Processing Systems**. Curran Associates, Inc., 2023. v. 36, p. 29343–29371. Available from Internet: <https://proceedings.neurips.cc/paper_files/paper/2023/file/5d9571470bb750f0e2325a030016f63f-Paper-Datasets_and_Benchmarks.pdf>.

HE, K. et al. Deep residual learning for image recognition. In: **Proceedings of the IEEE conference on computer vision and pattern recognition**. [S.l.: s.n.], 2016. p. 770–778.

JIN, X.; HAN, J. K-means clustering. In: ____. **Encyclopedia of Machine Learning**. Boston, MA: Springer US, 2010. p. 563–564. ISBN 978-0-387-30164-8. Available from Internet: <https://doi.org/10.1007/978-0-387-30164-8_425>.

KIMURA, M. Understanding test-time augmentation. In: MANTORO, T. et al. (Ed.). **Neural Information Processing**. Cham: Springer International Publishing, 2021. p. 558–569. ISBN 978-3-030-92185-9.

KIRILLOV, A. et al. **Segment Anything**. 2023. Available from Internet: <https://arxiv.org/abs/2304.02643>.

LECUN, Y. et al. Backpropagation applied to handwritten zip code recognition. **Neural Computation**, v. 1, n. 4, p. 541–551, 1989.

MOREIRA, T. P. et al. Where is my puppy? retrieving lost dogs by facial features. **Multimedia Tools and Applications**, v. 76, n. 14, p. 15325–15340, Jul 2017. ISSN 1573-7721. Available from Internet: <https://doi.org/10.1007/s11042-016-3824-1>.

MOUGEOT, G.; LI, D.; JIA, S. A deep learning approach for dog face verification and recognition. In: NAYAK, A. C.; SHARMA, A. (Ed.). **PRICAI 2019: Trends in Artificial Intelligence**. Cham: Springer International Publishing, 2019. p. 418–430. ISBN 978-3-030-29894-4.

OQUAB, M. et al. **DINOv2: Learning Robust Visual Features without Supervision**. 2024. Available from Internet: <https://arxiv.org/abs/2304.07193>.

PETSRS. **PetsRS: Encontre seu pet**. 2024. Available from Internet: <https://petsrs.com.br/>.

RADFORD, A. et al. Learning transferable visual models from natural language supervision. **CoRR**, abs/2103.00020, 2021. Available from Internet: <https://arxiv.org/abs/2103.00020>.

REDMON, J. et al. You only look once: Unified, real-time object detection. In: **2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)**. Los Alamitos, CA, USA: IEEE Computer Society, 2016. p. 779–788. ISSN 1063-6919. Available from Internet: <https://doi.ieeecomputersociety.org/10.1109/CVPR.2016.91>.

SCHROFF, F.; KALENICHENKO, D.; PHILBIN, J. Facenet: A unified embedding for face recognition and clustering. **CoRR**, abs/1503.03832, 2015. Available from Internet: <http://arxiv.org/abs/1503.03832>.

SELVARAJU, R. R. et al. Grad-cam: Visual explanations from deep networks via gradient-based localization. In: **2017 IEEE International Conference on Computer Vision (ICCV)**. [S.l.: s.n.], 2017. p. 618–626.

SUL, G. do estado do Rio Grande do. **Defesa Civil atualiza balanço das enchentes no RS – 9/6, 9h**. 2024. Available from Internet: <https://www.estado.rs.gov.br/defesa-civil-atualiza-balanco-das-enchentes-no-rs-9-6-9h>.

TERVEN, J.; CóRDOVA-ESPARZA, D.-M.; ROMERO-GONZáLEZ, J.-A. A comprehensive review of yolo architectures in computer vision: From yolov1 to yolov8 and yolo-nas. **Machine Learning and Knowledge Extraction**, v. 5, n. 4, p. 1680–1716, 2023. ISSN 2504-4990. Available from Internet: <https://www.mdpi.com/2504-4990/5/4/83>.

WANG, A. et al. Yolov10: Real-time end-to-end object detection. **arXiv preprint arXiv:2405.14458**, 2024.

YOON, B.; SO, H.; RHEE, J. A methodology for utilizing vector space to improve the performance of a dog face identification model. **Applied Sciences**, v. 11, n. 5, 2021. ISSN 2076-3417. Available from Internet: <https://www.mdpi.com/2076-3417/11/5/2074>.

ZAIDI, S. S. A. et al. A survey of modern deep learning based object detection models. **Digital Signal Processing**, v. 126, p. 103514, 2022. ISSN 1051-2004. Available from Internet: <https://www.sciencedirect.com/science/article/pii/S1051200422001312>.