Automatic Retouch Dog Eyes

Yi Ge
CS 280A Final Project, UC Berkeley
billy.ge@berkeley.edu

Abstract

While it is easy and popular to retouch a human being in photos, such as making the eyes larger or the body slimmer, none of those tools are applicable to dogs, because of the inability to recognize facial features of dogs. In this paper, we make it possible for dog-lovers to enlarge or shrink dog eyes in our favorite dog pictures. We accomplish the task by fine-tuning the YOLO object detection model to automatically identify dog eyes using the Columbia Dog Dataset. Then we apply the triangular mesh technique in image warping and morphing to make changes to the detected dog eyes regions based on user controls in real time.

1. Introduction

When we talk about machine learning and dogs, many research have centered around breed classification [1, 5, 6]. However, many cool tools which are accessible to humans (Figure 1) are not yet made available to dogs, such as automatic retouching body regions. Figure 2 shows the inability

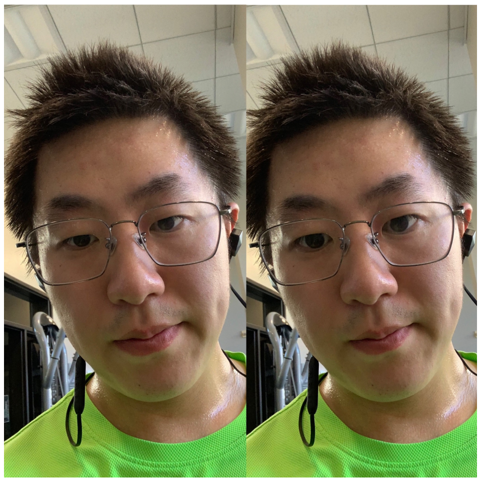
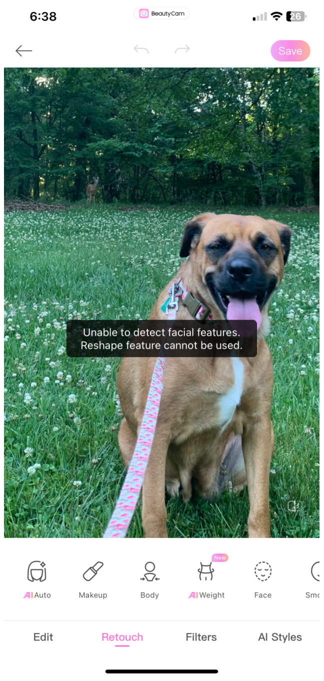


Figure 1. Example of human retouch effects. It includes enlargement of the eyes and shrinking of the mouth. It is done using BeautyCam.



Unable to detect facial features.
Reshape feature cannot be used.

Figure 2. Example of dog retouch fail on BeautyCam.

to recognize dog faces in a popular human photo editing platform called BeautyCam. If a human being shows up in an image, all we need to do is to select the body regions we want to retouch. A scrolling bar will then appear for users to control the degree in which he or she wants to change. Unfortunately, those tools are not currently available to dogs.

While it is our future goal to allow equal access of automatic retouching of all body parts to dogs, this paper presents the very first effort to establish a solution pipeline using dog eyes as our starting point.

At a high level, our solution is made up of two parts. First, we enable automatic dog eyes detection by fine-tuning the popular YOLO (You Only Look Once) object detection model [2, 4]. Second, we expand or contract the detected dog eye bounding boxes based on user control to perform image warping and morphing using the triangular

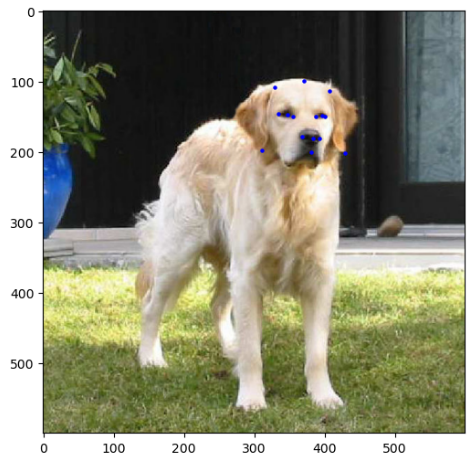


Figure 3. Example of fifteen dog facial feature manual labels in Columbia Dog Dataset.

mesh technique. More details about our methods are explained in the methodology section.

2. Data

The dataset we use in this paper is the Columbia Dog Dataset, which contains 8350 dog images of 133 different dog breeds. In addition, there are fifteen manually-labeled dog facial features in each dog picture.

In our paper, the data are mainly used for fine-tuning the YOLO model. Among the fifteen manually-labeled dog facial features, six are relevant to dog eyes in which we take advantage of. An example of the manual labeling is shown in Figure 3.

In order to balance model performance and computation costs, we did not use the entire Columbia Dog Dataset. Instead, we randomly sampled 1330 dog images for training and 665 dog images for validation. In addition, our sampled data comes from the 133 dog breeds evenly, so that our fine-tuned model share the same performance in all dog images.

3. Methodology

There are two stages in our method. First is fine-tuning an object detection model. Second is enlarging or shrinking the detected dog eyes based on user control.

In order to enable automatic detection of dog eyes, we decide to begin with the YOLO (You Only Look Once) model, due to its speed, accuracy, and easy-to-customize environment [4]. YOLO is a CNN-based object detection model, trained on the COCO dataset (Common Objects in Context) [3] with 80 classes. Interestingly, the pre-trained YOLO model is able to detect dogs, since dog is one of the 80 classes in the COCO data. However, it is not able to detect dog eyes. The YOLO model made its name in 2016 by Redmon *et al*. Over the past ten years, many improvements have been added, making the popular model more powerful. The version of YOLO model we are using in this paper is YOLO 11n, which was released as recent as September, 2024 [2]. Figure 4 shows the performance comparison between different YOLO model versions.

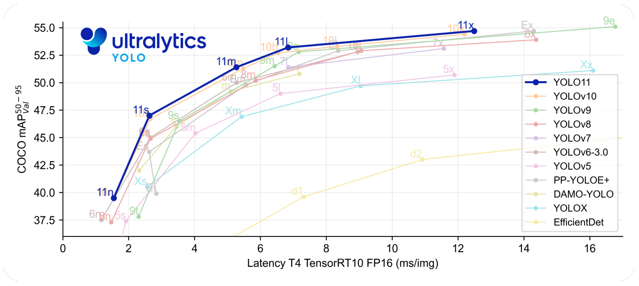


Figure 4. Performance comparison among different YOLO model versions.

To fine-tune the pre-trained YOLO model, we take advantage of the manually labeled dog images in the Columbia dog dataset. There are three manual labels for each dog eye. One on the left of the dog eye, one on the right of the dog eye, and one in the middle. However, the ground truth for fine-tuning has to be a bounding box, which means we need not only the two sides of the dog eye, but also the top and bottom. To solve the problem, we decide to first take the min and max both horizontally and vertically in each dog eye three-manual-labels. Second, we pick scaling factors conservatively to multiply the horizontal min by 0.98, the horizontal max by 1.02, the vertical min by 0.93 and the vertical max by 1.07. We believe that the two sides of the dog eye have accurate labels, hence we choose not to scale it aggressively. YOLO environment requires all custom dataset used for fine-tuning the model to be in a particular format, so we prepare our sampled Columbia Dog Dataset accordingly.

After data preparation, we fine-tune the YOLO model

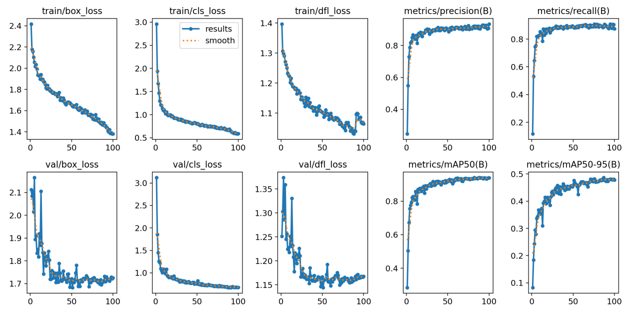


Figure 5. Change in performance over 100 epochs of fine-tuning YOLO.

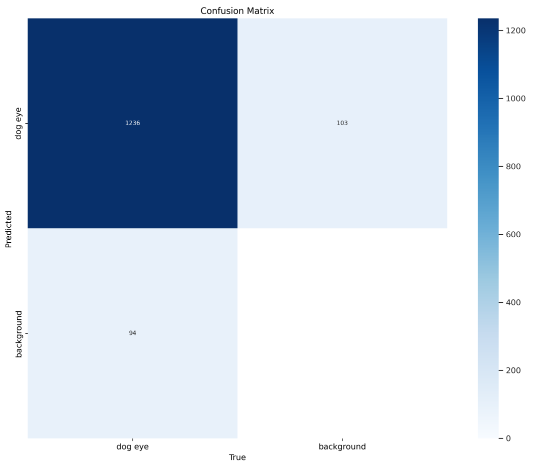
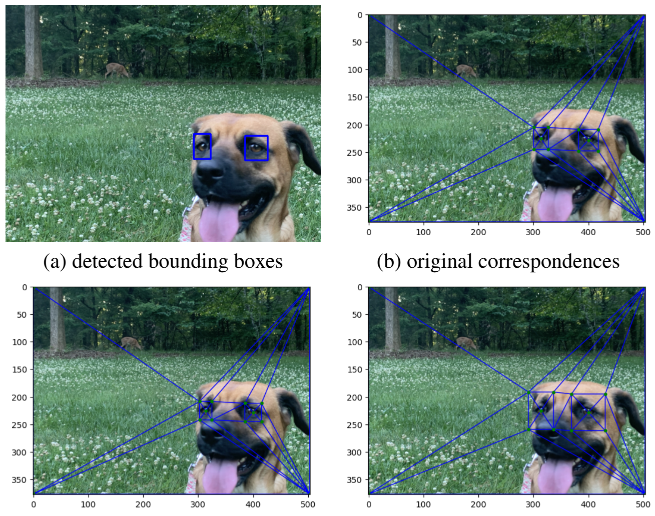


Figure 6. Confusion matrix result in the training data post fine-tuning.


(a) detected bounding boxes

(b) original correspondences

(c) shrinking correspondences

(d) enlarging correspondences

Figure 7. Example of dog eye retouch steps.

for 100 epochs, which takes less than an hour on T4 GPU. Figure 5 tracks the change in performance in the fine-tuning process. It is obvious that most metrics have plateaued, but training deeper may still improve the model although it may not be cost efficient. Figure 6 shows the performance of our fine-tuned YOLO model on the training data. It is able to correctly detect dog eyes more than 90% of the time.

Figure 7a shows the output of our fine-tuned YOLO model. Two dog eyes are detected, represented in blue bounding boxes. The next step is to retouch dog eyes based on user control of enlarging or shrinking effects. We accomplish this with the triangular mesh algorithm.

In triangular mesh, we first collapses all bounding box coordinates and put them into a correspondence point list. Then for each bounding box, we add the center point of the bounding box to our list. We also add the four corner points of the image to the list. Now we perform Delaunay triangulation. Figure 7b shows the example output after Delaunay triangulation.

Based on user control, we set the scaling factor to expand or contract the bounding boxes for each dog eye. We follow the implementation on BeautyCam, where users are able to at most scale up or down the size by 2. We implement this by asking the user to enter a value between 1 and 100, assuming the current size is 50. We then replace the bounding box coordinates in our original correspondence list with the new bounding box coordinates, and plotting using the same triangulation. Figure 7c shows the shrinking bounding boxes with scaling factor equals 35 whereas Figure 7d shows the enlarging bounding boxes with scaling factor equals 85.

Next, we morph the original correspondences to the adjusted correspondences using the same triangulation. We do so by iteratively going through each triangle. First, we compute the affine transformation matrix. Second, we use the inverse of the affine transformation matrices to interpolate color for each pixel inside a triangle.

4. Results

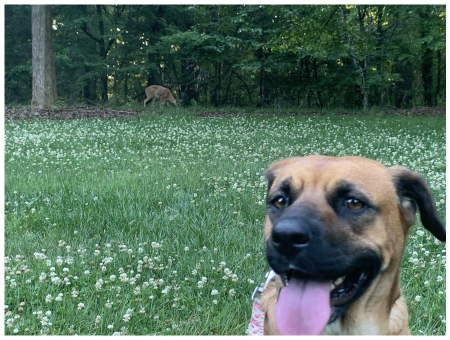
Figure 8 shows the final outputs of the example in Figure 7. It is pretty obvious that Figure 8b has smaller dog eyes and Figure 8c has larger dog eyes compared to the original input dog image. Everything looks pretty natural.

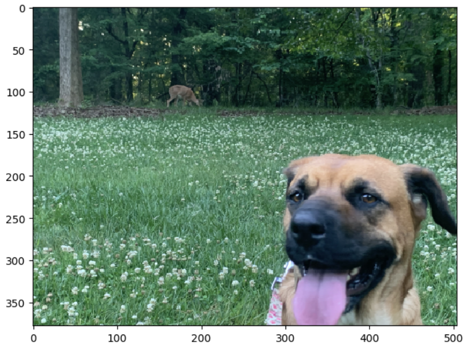
More visual examples are shown in Figure 10 and Figure 11 in the appendix.

5. Limitation

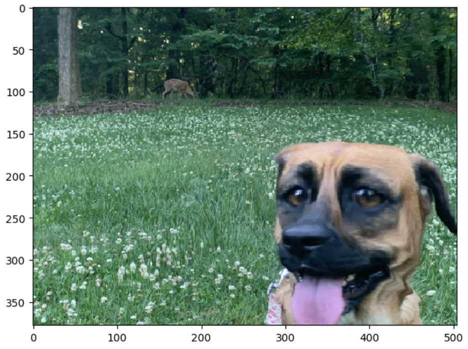
There are two main limitations in our approach, one about the fine-tuned YOLO model not able to automatically detect all dog eyes in an image, the other about visual defects in the triangular mesh output.

First, our fine-tuned YOLO model is not able to detect all dog eyes in the image, as shown in Figure 9, although it is able to distinguish dog eyes from human eyes and other animal eyes (Figure 12 in the appendix). When there are more than one dog in the input image, our fine-tuned YOLO model fails most of the time. The main reason behind this problem is our training data. Although the Columbia Dog Dataset contains manually labeled dog facial features in each dog image, it only labels the most dominant dog, even though there may be more than one dogs in the image. Taking advantage of this incomplete manual labeling, we decide to ignore this data issue in order to save ourselves a lot of hard labor work, leading to the fine-tuned YOLO model maybe thinking that there cannot a lot of dog eyes in one input dog image.


(a) original input image



(b) shrunk dog eyes



(c) enlarged dog eyes

Figure 8. Example of dog eye retouch results.

Second, our triangular mesh output has a clear defect: it does not only changes the size of the dog eyes, it also changes the relative size of every object in the image, including the dog. Please take a look at Figure 10 and Figure 11 in the appendix for more visual examples. The direct cause of this issue is that there are too few correspondence points when computing our triangulation. In the current approach, many bounding box corner point form large triangles with the corner points of the image. Therefore, when the triangle changes, objects outside the dog also changes. One simple solution is to add four more coordinates slightly outside the bounding box for each dog eye detected. In this way, the outside bounding box is connected to the corner points of the image. Those triangles are not changed in the interpolation process. The inner triangles change due to changes in bounding boxes based on user control. These triangle changes are limited on dog faces.

6. Conclusion

This paper shows that it is not difficult to apply the cool visual tools that human can enjoy to dogs. Although our methodology have some defects, the overall visual effects are natural and fun. We look forward for future works in related fields!

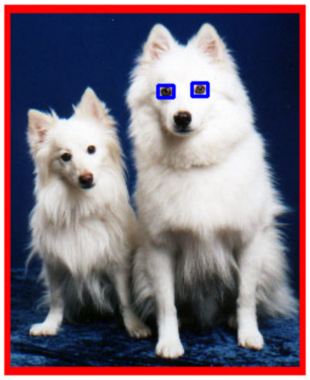
References

[1] Ying Cui, Bixia Tang, Gangao Wu, Lun Li, Xin Zhang, Zhenglin Du, and Wenming Zhao. Classification of dog breeds using convolutional neural network models and support vector machine. *Bioengineering*, 11(11), 2024. 1

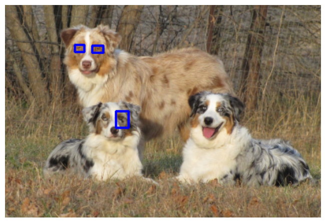
[2] Rahima Khanam and Muhammad Hussain. Yolov11: An overview of the key architectural enhancements, 2024. 1, 2

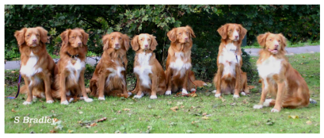
[3] Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, and Piotr Dollár. Microsoft coco: Common objects in context, 2015. 2

(a)


(b)

(c)


(d)


(e)

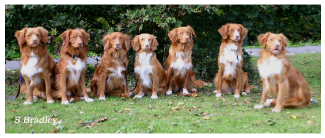

(f)

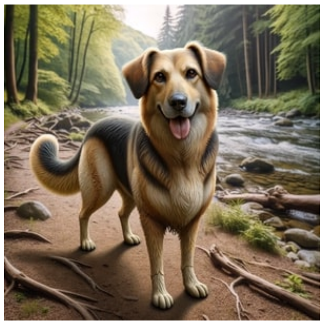
Figure 9. Example of YOLO dog eye detection fails.

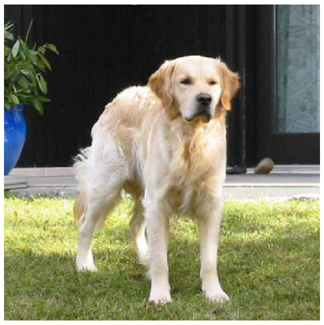
[4] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection, 2016. 1, 2

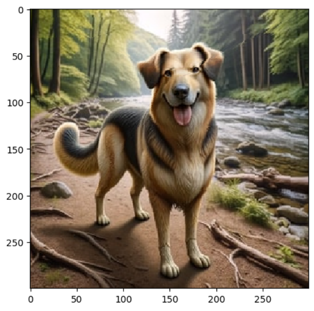
[5] Nattakan Towpunwong and Napa Sae-Bae. Dog breed classification and identification using convolutional neural networks. *ECTI Transactions on Computer and Information Technology (ECTI-CIT)*, 17:554–563, 12 2023. 1

[6] Akash Varshney, Abhay Katiyar, Aman Singh, and Surendra Chauhan. Dog breed classification using deep learning. pages 1–5, 06 2021. 1

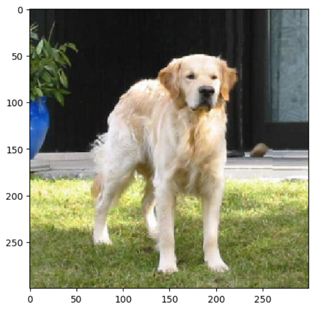
7. Appendix


(a) original input image

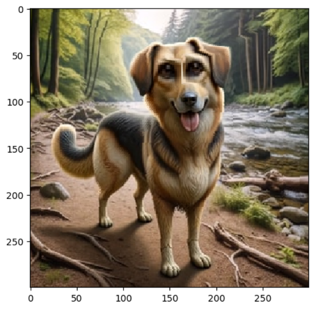

(b) original input image



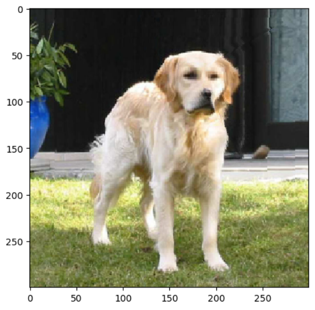
(c) scaling factor = 10



(d) scaling factor = 35



(e) scaling factor = 90

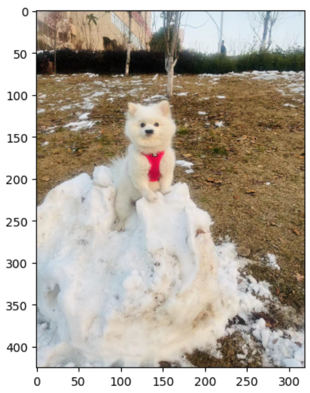


(f) scaling factor = 70

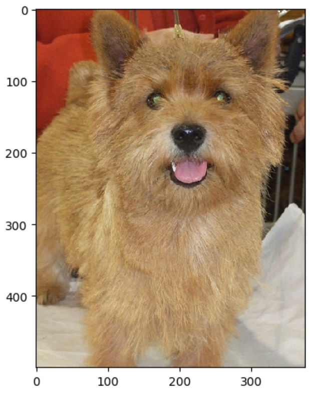
Figure 10. More visual examples of dog eyes shrinking and enlarging effects.

(a) original input image

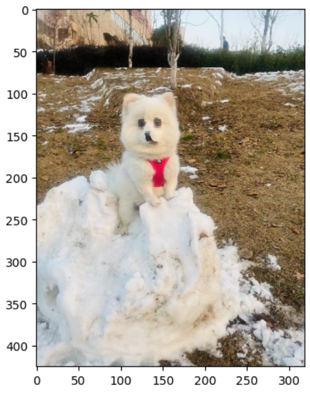
(b) original input image



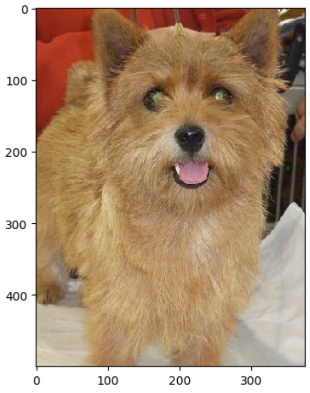
(c) scaling factor = 30



(d) scaling factor = 40

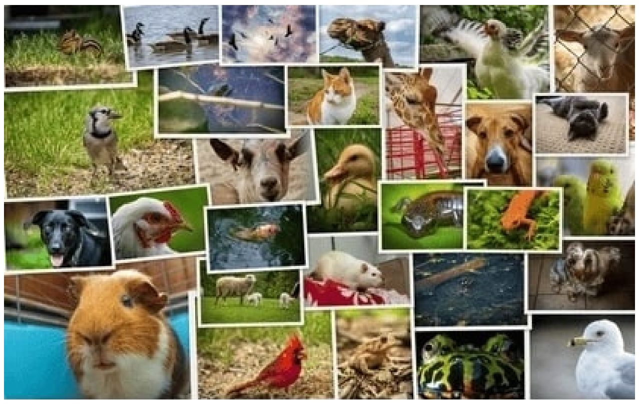


(e) scaling factor = 80



(f) scaling factor = 60

Figure 11. More visual examples of dog eyes shrinking and enlarging effects.


(a) original input image

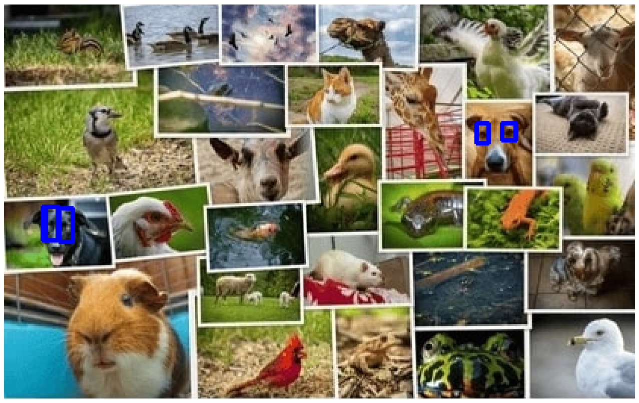

(b) original input image

Figure 12. Our fine-tuned YOLO model is able to distinguish dog eyes from other animal eyes.