Milestone 3 Team Data Pets Documentation

Ivan Shu, Xiang Bai, Sean Gao, Benjamin Liu, Yuxin Xu

Background

Austin Pets Alive (APA) is an association of pet owners with a database of available pet photos. The goal will be to build a reusable application, design, and framework that can be implemented in any animal welfare nonprofits to connect future pet owners with pets. The outcome of the project will be to build a full featured application for the APA.

Problem Definition and Project Scope

For this Pets project, our group's focus will be to create a user friendly tool to match potential dog loving adopters/owners to dogs available for adoption. As stated above, the end goal is developing a nice web application that can harness the data and be used with features/deliverables that will be helpful for the matching process.

The core problem we are trying to solve is to help future dog owners find a dog who is a good fit for their lifestyle and family environment. First we help the user search for dogs based on certain features such as size, color, and breed or the images users upload. Secondly we will connect the dog with the user by allowing the user to chat with a persona of the dog. The user can ask this virtual dog any question about it -- its breed characteristics, or any general questions about puppies and dogs.

Data Science Pipeline Logistics

All the data is stored in google cloud platform, and models will be run and trained in Colab. We plan to containerize both front-end and back-end applications using docker and deploy the app using Kubernetes (more details are to be added in milestone 3). To accelerate training process, we resized the images, created TFRecords, generated predicted embeddings and stored them in our GCP bucket.

Exploration Data Analysis

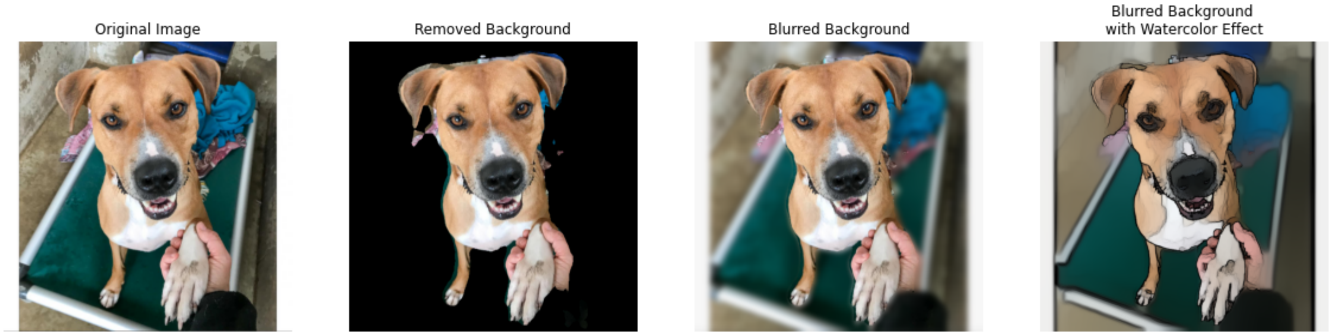
APA makes available a repository for its animals that is roughly ~17k dog records and close to 40k if you include cats. For the ~40k pets there looks to be ~140k photos. Our dataset consists of several csv files [Dataset Link], including dogs metadata, dogs images, website memos, as well as dogs questions and answer datasets.

Proposed Solution I: Computer Vision

To help the matching process, we leverage several computer vision models to accomplish a few specific tasks. As an adoption center, when adopting new dogs from users, a function will be available for some pre-processing of the images.

- Remove noisy background from the uploaded dog picture using DeepLabv3 plus.
- Allow to choose and add new backgrounds/effects.
- Enhance the image if the solution of the uploaded picture is not ideal.

1. Removing Old and Add New Background with Different Effects



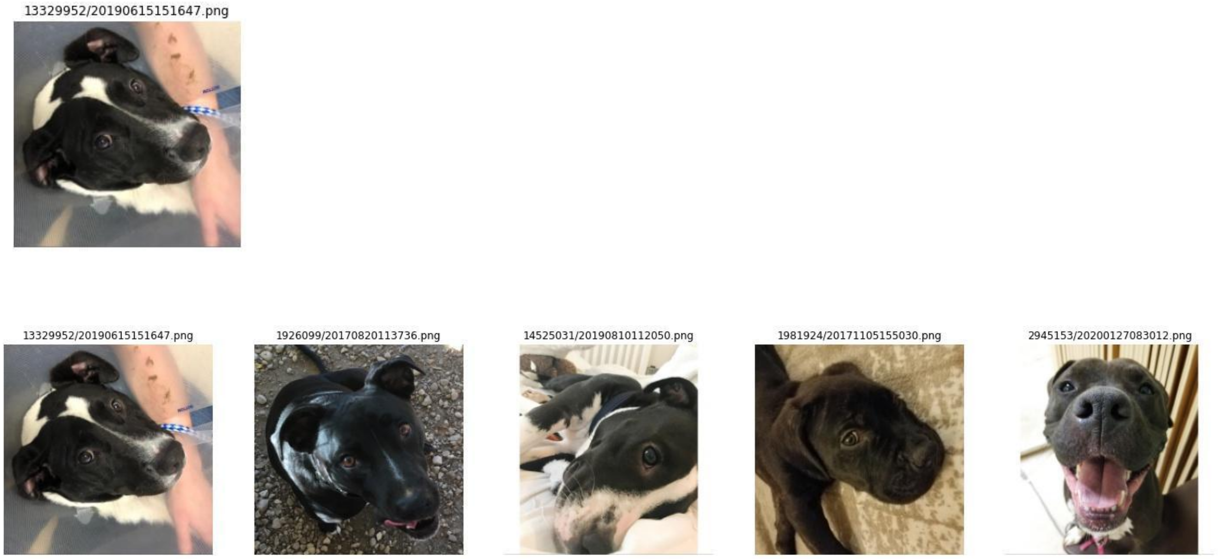
One the very left shows the original image, and the second image was after the background was removed by DeepLabv3 Plus. The third and forth images are different backgrounds users or adoption sites can choose when donating or adopting new dogs.

On the other hand, to allow for users searching for their interested dogs, some other tasks listed below need to be considered.

- Create embeddings using EfficientNet for all the dog images in the dataset.
- Search top 5 similar images using Facebook AI Similarity Search (FAISS).
- Segment and remove background before embedding search by using DeepLabv3 plus.

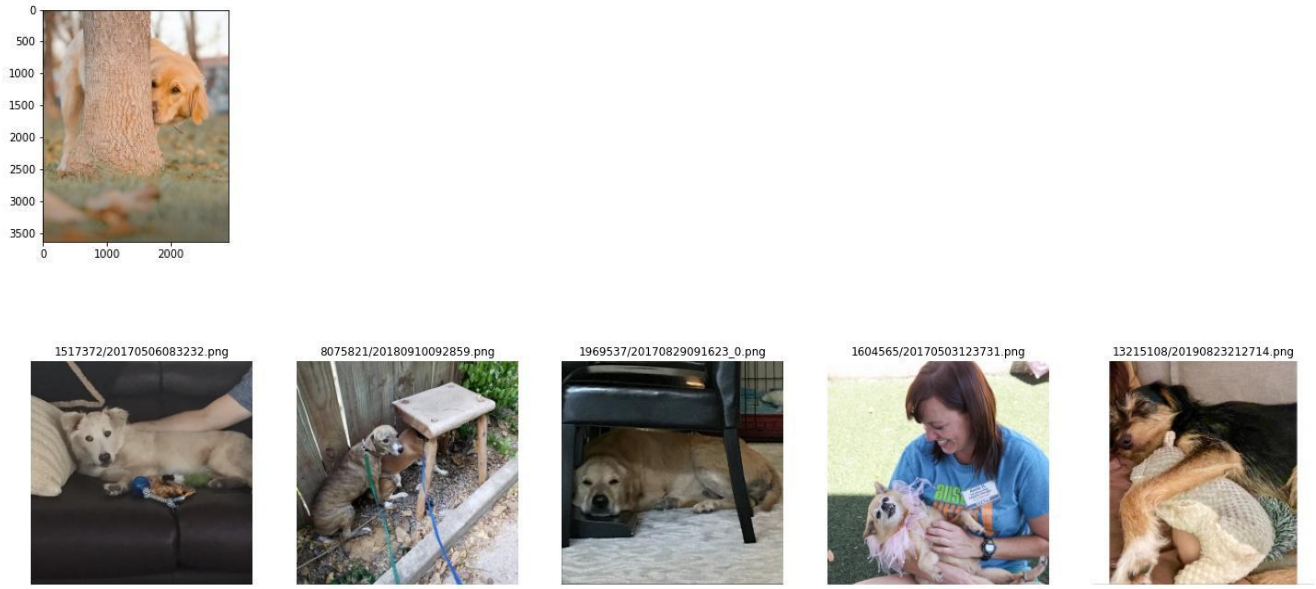
One way to search similar images given an input is to create image embeddings and look for similar ones in that space. To generate embeddings, we used EfficientNet, B0 model [1], as shown to have better performance with less parameters. Then we used an embedding search algorithm developed by Facebook, FAISS to find the top 5 similar images [2]. See figure 2 for some results.

Figure 2. Example Matched Images By Using EffecientNet and FAISS Embedding Search


13329952/20190615151647.png
13329952/20190615151647.png
1926099/20170820113736.png
14525031/20190810112050.png
1981924/20171105155030.png
2945153/20200127083012.png

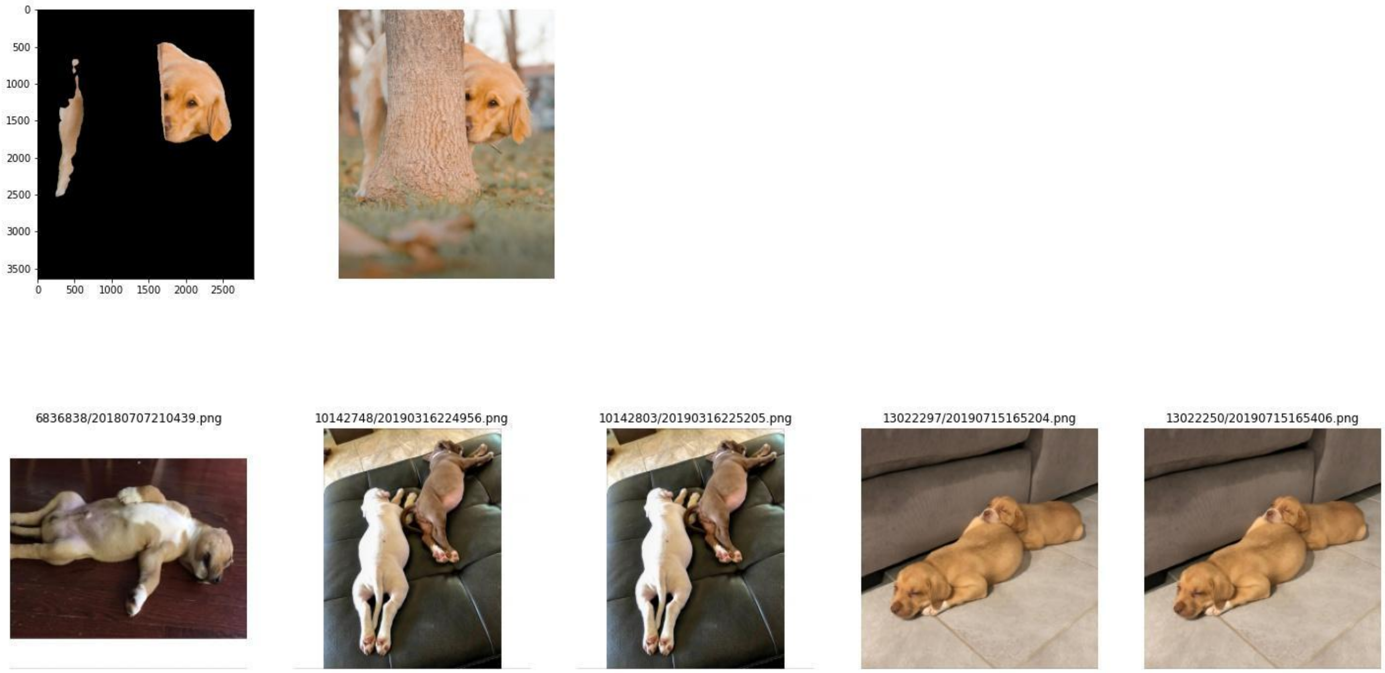
The top row shows the input image and the second row shows the top 5 matched images using FAISS. However, one problem we encountered is that when generating embedding for an input image that has noises or features not related to dogs, the matched images contain irrelevant features as well. See figure 3. Note: the main steps of model training were referenced by Rashmi's notebook during 2021 ComputeFest.

Figure 3. Example Input Images That Contains Dog-Irrelevant Features


0
500
1000
1500
2000
2500
3000
3500
0
1000
2000
1517372/20170506083232.png
8075821/20180910092859.png
1969537/20170829091623_0.png
1604565/20170503123731.png
13215108/20190823212714.png

Here, we can see that when the inputted image has a large dog-irrelevant feature (tree), the matched images contain dog-irrelevant features as well. For example, the matched images contain arms, chairs or humans. This causes the searched pictures to contain less information about the dog, which led to unideal results. To bypass this constraint, we introduced another pre-trained model, DeepLabv3 plus to pre-segment the image before embedding search [3].

Figure 4. Pre-segment Input Images Before Embedding Search Similar Images

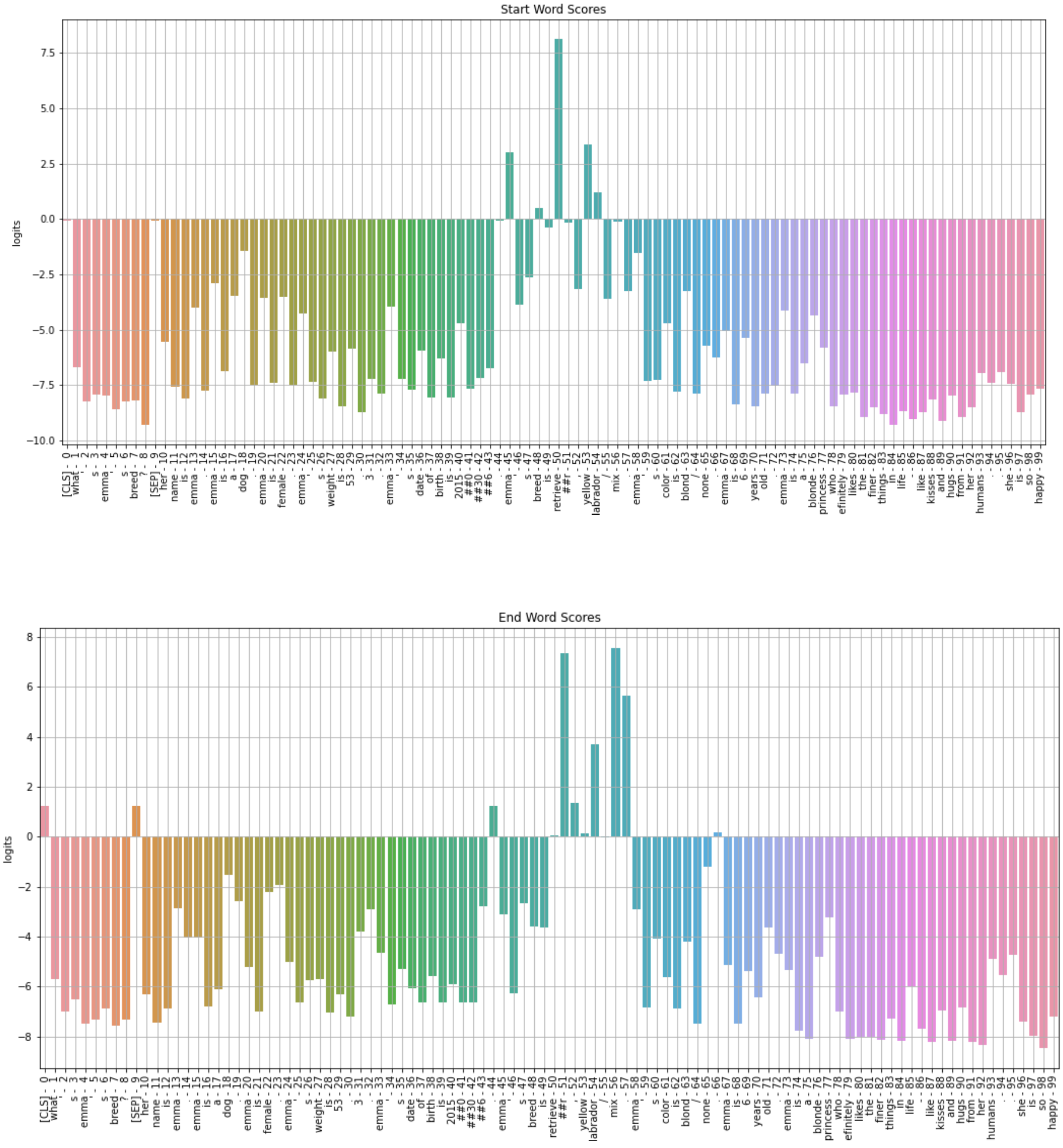

0
500
1000
1500
2000
2500
3000
3500
0
500
1000
1500
2000
2500
6836838/20180707210439.png
10142748/20190316224956.png
10142803/20190316225205.png
13022297/20190715165204.png
13022250/20190715165406.png

After the image was segmented and the background was removed, the matched images contained more features or information related to dogs, such as age, or fur color. One caveat is that the matched images might contain multiple dogs because of the segmentation.

Proposed Solution II: Natural Language Processing

The next task is to create a persona of the dog that the user has searched and clicked and allow users to ask more information regarding the dog by directly chatting with the persona. To implement a baseline model, we use the BERT question and answering model, which was trained on the Stanford Question Answering Dataset (SQuAD) [4]. We will provide sample questions and reference text to the model and let the model predict the start and end token of a “span” of text which acts as the output answer. We followed the main steps from Shivas’ notebook in 2021 ComputeFest [5] and the tutorial by Chris McCormmick [6].

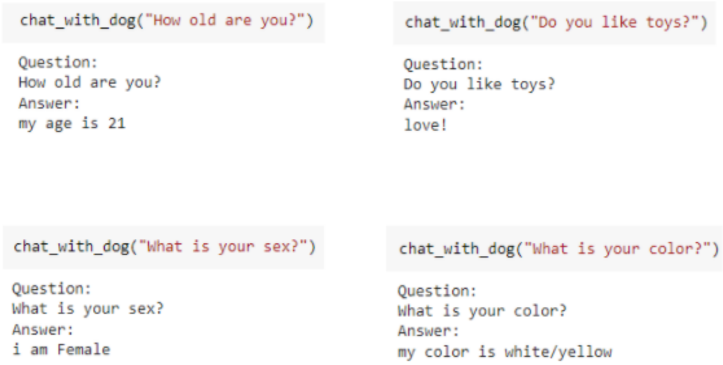
Figure 5. Probability scores for start and end tokens predicted by BERT with an example question



In one example, we asked the question, “what’s Emma’s breed” and fed it together with a reference text. The model correctly predicted with the correct span of text in the reference, which is "retriever , yellow labrador / mix". To further visualize the probabilities of each start and end tokens, we plotted the results in Figure 5. Both “retriever” and “mix” are predicted with highest probabilities for either start and end tokens.

However, one major drawback about BERT is it answers the question by providing a span of text in the reference text and this prevents forming a natural conservation like human-to-human interaction between users and the app. To circumvent this problem, we then decided to try generative-based language models, GPT2 for question and answering tasks. We fine-tuned the GPT2 model with specific context on which the dogs were based. The resulting Q&A outputs are quite good, and we displayed some of the examples in Figure 6.

Figure 6. Example output from fine-tuned GPT2 model on dog “Emma”



```
chat_with_dog("How old are you?")

Question:
How old are you?
Answer:
my age is 21
```

```
chat_with_dog("Do you like toys?")

Question:
Do you like toys?
Answer:
love!
```

```
chat_with_dog("What is your sex?")

Question:
What is your sex?
Answer:
i am Female
```

```
chat_with_dog("What is your color?")

Question:
What is your color?
Answer:
my color is white/yellow
```

Next Step

We are planning on finalizing our deployment of natural language processing features and diving into app design for the following week. Specifically, we will add the question and answering feature to Docker containers, deploy it to the GCP using Kubernetes and start exploring Node JS for the front-end.

Acknowledgment

Many model trainings in this project were based on the 2021 Harvard IACS ComputeFest [7] for our learning purpose. We want to thank Shivas, Rashmi for their example codes on NLP and CV tasks. Especially, we want to thank our project TF, Shivas, for his time, help, and guidance in the whole process of completing this project.

Timeline for the remaining project

- Week 12/01- 12/07
    - Project Wrap Up: NLP containerization and React deployment

Reference

1. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks, Mingxing Tan et al.

2. FAISS, Facebook AI Similarity Search.

3. DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs, Liang-Chieh Chen et al.

4. SQuAD, Stanford Question Answering Dataset.

5. 2021 Harvard IACS ComputeFest Computer Vision Task Notebook.

6. Question Answering with a Fine-Tuned BERT, Chris McCormick.

7. 2021 Harvard IACS ComputeFest GitHub Repository.