DOG BREED RECOGNITION CHATBOT

Jason Cole

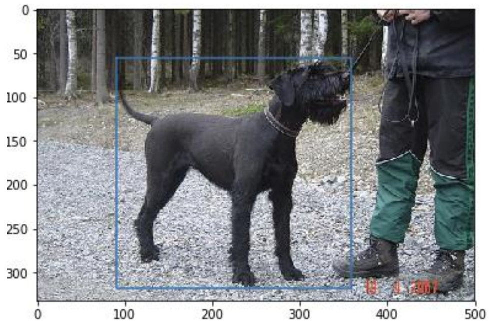
The intent of the project was to create an entertaining dialogue system thought as a dog character that it would also be able to recognize the breed of a dog when the user would pass an image through. For this reason the project is divided in two parts: on one side the image recognition part is developed, on the other one is displayed the chatbot.

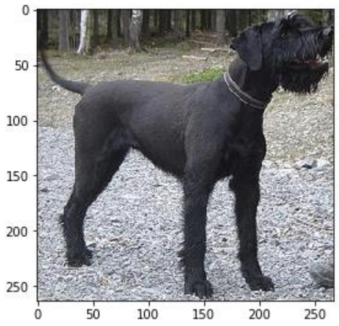
Link of github: https://github.com/jjccooooll12/DogBot/

IMAGE RECOGNITION

The dataset of the image recognition is the Stanford Dogs Dataset and can be found at http://vision.stanford.edu/aditya86/ImageNetDogs/main.html. The parts of the dataset downloaded comprised the images (20,580 images) and the annotations. In the annotations were found the labels of all the images (120 different kind of breeds) and also the values of the bounding boxes to crop the pictures at just the size of the dog, so to have less sparsity during training.

The first action, then, has been to create a function to crop all the pictures, using ElementTrees. The Crop.ipynb notebook does just that





Then, all the cropped pictures have been saved in a new file in order to work directly from there.

In the Model.ipynb I extracted the cropped images from the new folder, split the data in training, validation and test set (training 80%) and then preprocessed the images by augmenting them in the training set. Augmentation involves random rotations, resize, cropping, flip and randomly changing the brightness, contrast and saturation of the images. The images are then normalize using the Imagenet standards and transformed to tensors. In validation and test set I didn't augment the pictures, just adapted them to the desired size, transformed to tensor and normalized.

For the model, I wanted to try to make my own CNN and see how it goes. After trying with different layers, hyperparamaters and so on unfortunately the accuracy didn't go over 66% in validation. For this reason I turned to a pretrained model. I tried first with Alexnet but didn't have a good result neither, I tried lastly with Inception and the accuracy was good enough to continue the experiment with this model.

On top of the pretrained model, I added a linear layer to project to the desired amount of outputs (120) and I configured it so that the early layers would freeze. At the end I applied a SoftMax function to get a probability score.

Running the model for about 10 epochs (I am not sure about the total number because I stopped in between and continued for other 4 epochs), the results have been satisfying:

Training Loss: 0.5949

Validation Loss: 0.7368

Training Accuracy: 81.60%

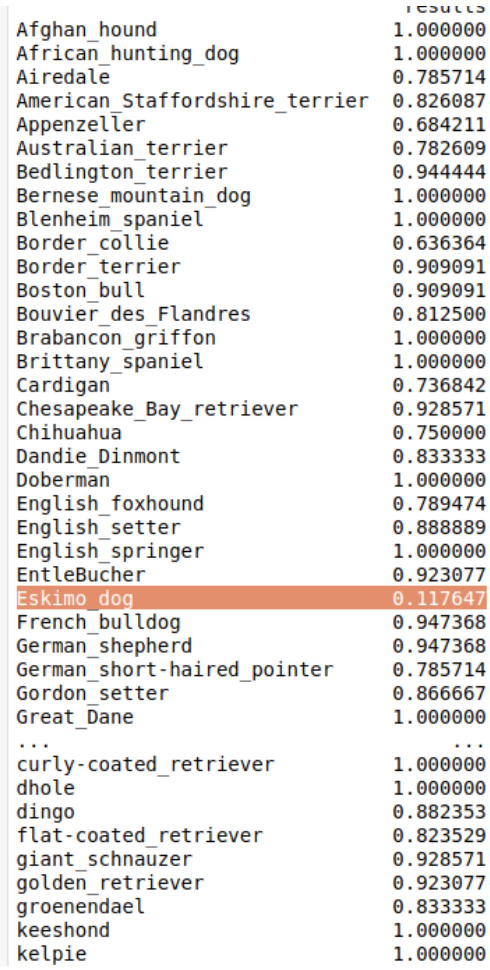
Validation Accuracy: 78.38%

Then I gave one shot on the Test set:

The average loss is 0.3640129846685073

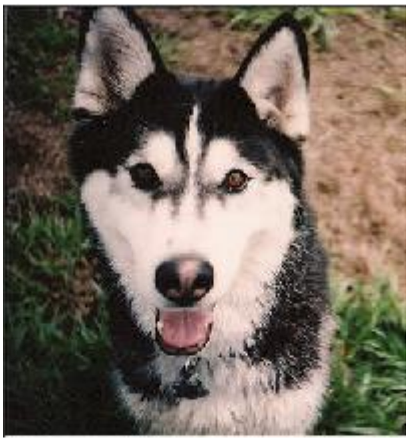
Accuracy: 1845/2058 (89%)

I tried to analyze the results but being 120 classes it was hard to have a clean idea about which were the general traits that made some breeds to performe well or not. However, I can notice that there is especially one class that underperformed in comparison to the overall results, which is Eskimo dog with only 11% of accuracy.

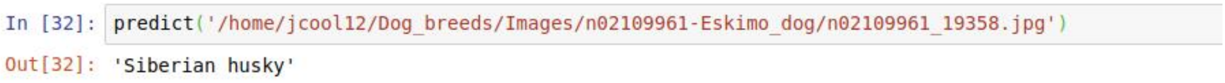


```
                                  results
Afghan_hound                     1.000000
African_hunting_dog              1.000000
Airedale                         0.785714
American_Staffordshire_terrier   0.826087
Appenzeller                      0.684211
Australian_terrier               0.782609
Bedlington_terrier               0.944444
Bernese_mountain_dog             1.000000
Blenheim_spaniel                 1.000000
Border_collie                    0.636364
Border_terrier                   0.909091
Boston_bull                      0.909091
Bouvier_des_Flandres             0.812500
Brabancon_griffon                1.000000
Brittany_spaniel                 1.000000
Cardigan                         0.736842
Chesapeake_Bay_retriever         0.928571
Chihuahua                        0.750000
Dandie_Dinmont                   0.833333
Doberman                         1.000000
English_foxhound                 0.789474
English_setter                   0.888889
English_springer                 1.000000
EntleBucher                      0.923077
Eskimo_dog                       0.117647
French_bulldog                   0.947368
German_shepherd                  0.947368
German_short-haired_pointer      0.785714
Gordon_setter                    0.866667
Great_Dane                       1.000000
...                                   ...
curly-coated_retriever           1.000000
dhole                            1.000000
dingo                            0.882353
flat-coated_retriever            0.823529
giant_schnauzer                  0.928571
golden_retriever                 0.923077
groenendael                      0.833333
keeshond                         1.000000
kelpie                           1.000000
```

Eskimo dog



The reason I think is quite straightforward: it looks just like an Huski. As a matter of fact, if I try to predict the picture of the Eskimo, the model tells me that he’s sure of over 70% that this is a Siberian Husky



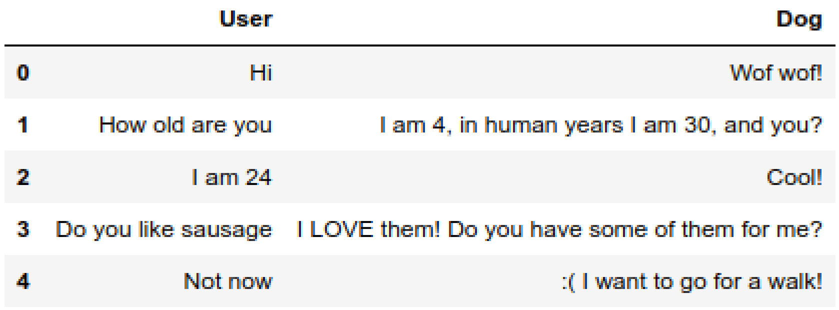
```
In [32]: predict('/home/jcool12/Dog_breeds/Images/n02109961-Eskimo_dog/n02109961_19358.jpg')
Out[32]: 'Siberian husky'
```

Moreover, the training pictures for the husky were 192 vs 150 of the Eskimo dog, this is most likely the reason why the model dropped so much in accuracy with this breed: because the contiguoos features and similarities with other classes advantaged as well by a wider training sample were too strong.

CHATBOT

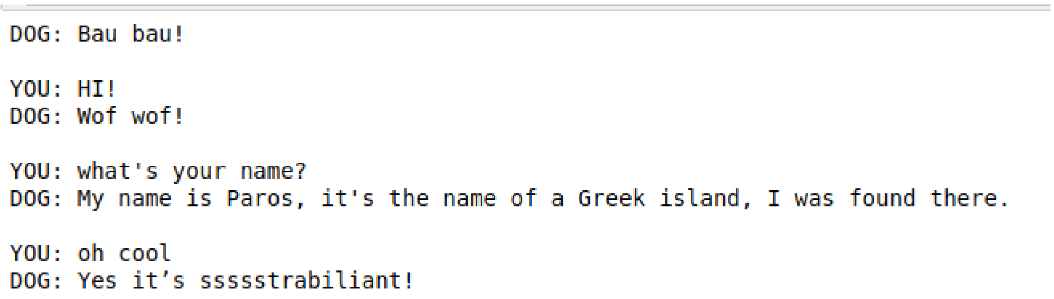
The file is Chatbot-predict.ipynb

For the chatbot I would have liked to build a Seq2Seq model, however, of course, there is no dataset of conversations between a human and a dog. For this reason I built a retrieval conversational agent creating the dataset myself through Pandas. In the dataframe, on one side there were questions and on the other one answers. I manually set around 30 question-answers at the beginning and built the dataframe of the chatbot.



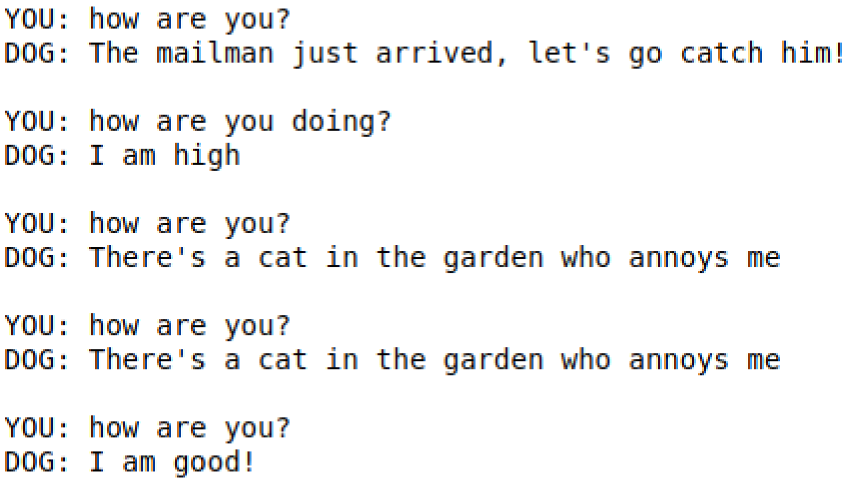
	User	Dog
0	Hi	Wof wof!
1	How old are you	I am 4, in human years I am 30, and you?
2	I am 24	Cool!
3	Do you like sausage	I LOVE them! Do you have some of them for me?
4	Not now	:(I want to go for a walk!

The chatbot starts with a while loop that breaks only in case the user types ‘bye’ or a picture is provided. As soon as the user types a question, the program reads the dataset, through Sklearn tf-idf fits question and answers and transforms the questions. Next, it fits and transforms the user input. At this point, cosine similarity between the input and the all the questions is computed: the question that retrieves the highest score returns the correspondant answer.



```
DOG: Bau bau!

YOU: HI!
DOG: Wof wof!

YOU: what's your name?
DOG: My name is Paros, it's the name of a Greek island, I was found there.

YOU: oh cool
DOG: Yes it's sssssstrabiliant!
```

The system is refined with some hard coding to ensure a little bit of more control to the answers. For example, if a digit is found in the query, then the asnwer will just be ‘Cool!’, on the other side if the question is matched with the entry ‘How are you?’ the program picks randomly between 4 different answers to make the bot more interesting.

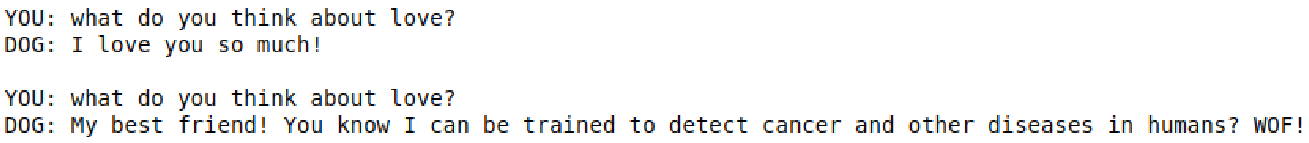


```
YOU: how are you?
DOG: The mailman just arrived, let's go catch him!

YOU: how are you doing?
DOG: I am high

YOU: how are you?
DOG: There's a cat in the garden who annoys me

YOU: how are you?
DOG: There's a cat in the garden who annoys me

YOU: how are you?
DOG: I am good!
```

This system can be easily expanded to many other queries in order to make the tool more enjoyable and fun.

When a question is not recognized with an identical matching by the program, then the system updates the csv file where the dataset is stored appending the new question and providing it with a defult ‘interesting fact’ that it picks randomly within the available ones. This mechanism assures two things:

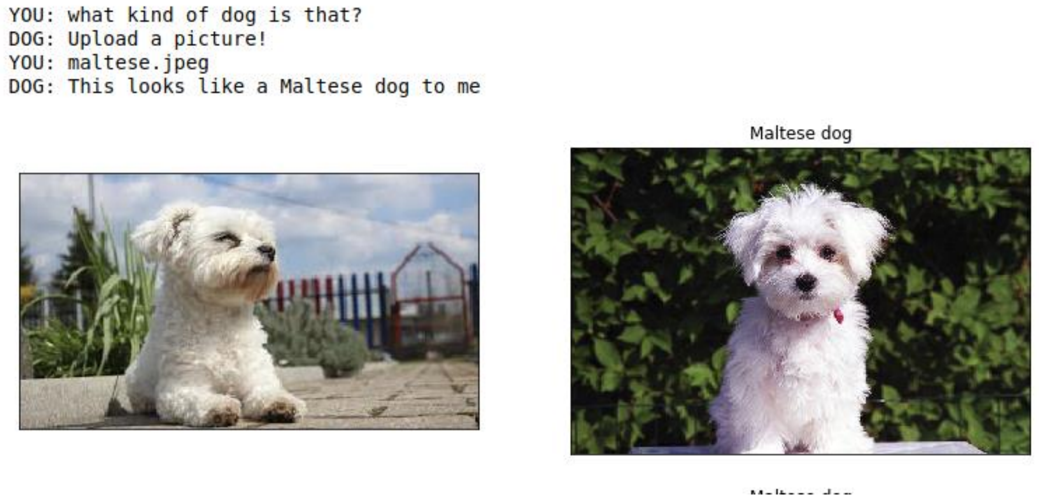
- All the queries from the users are registered so that in the future the answers can be updated and the tool has more data to perform better!
- If an user asks again the same question, he won’t return the same answer (which is boring to receive), but he will be surprised with a different response.



```
YOU: what do you think about love?
DOG: I love you so much!

YOU: what do you think about love?
DOG: My best friend! You know I can be trained to detect cancer and other diseases in humans? WOF!
```

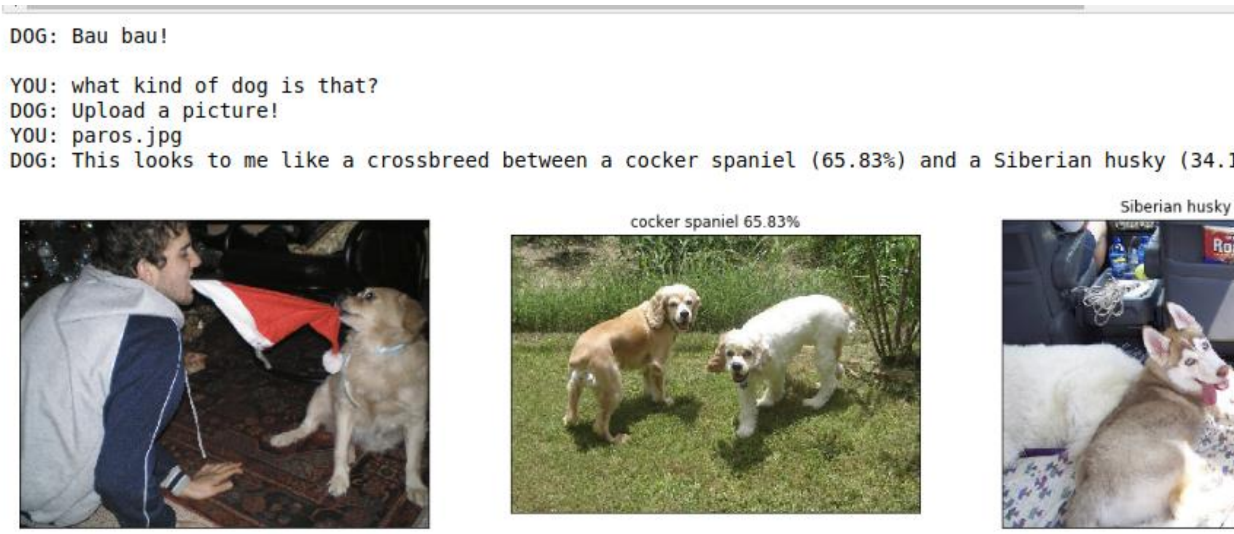
Adding at each turn the questions, it enabled me to pass from 30 question-answers to 200 in a couple of hours. Of course, the more queries we receive and the better it is. Also, once that a good amount of data is received we can even try to build a generative chatbot out of it.

Finally, I implemented in the bot a predict function that loads the parameters of the image recognition and, given a picture, computes the breed classification. The function shows also a picture of comparison between the image uploaded and the one predicted. If the program is sure about the breed of the dog (over a cutoff of 70%), then the answer states just the breed predicted.



```
YOU: what kind of dog is that?
DOG: Upload a picture!
YOU: maltese.jpeg
DOG: This looks like a Maltese dog to me
```

Otherwise, the chatbot retrieves the first two predictions and summing up their percentage and dividing them by the total I am passing the dog to be a crossbreed in percentage by two breeds (It’s not really true, but the user doesn’t know that :p) For example, this is the picture of my own dog which is indeed a crossbreed (a bastard) and we don’t know the origins. This is the prediction of the model when he’s not sure:



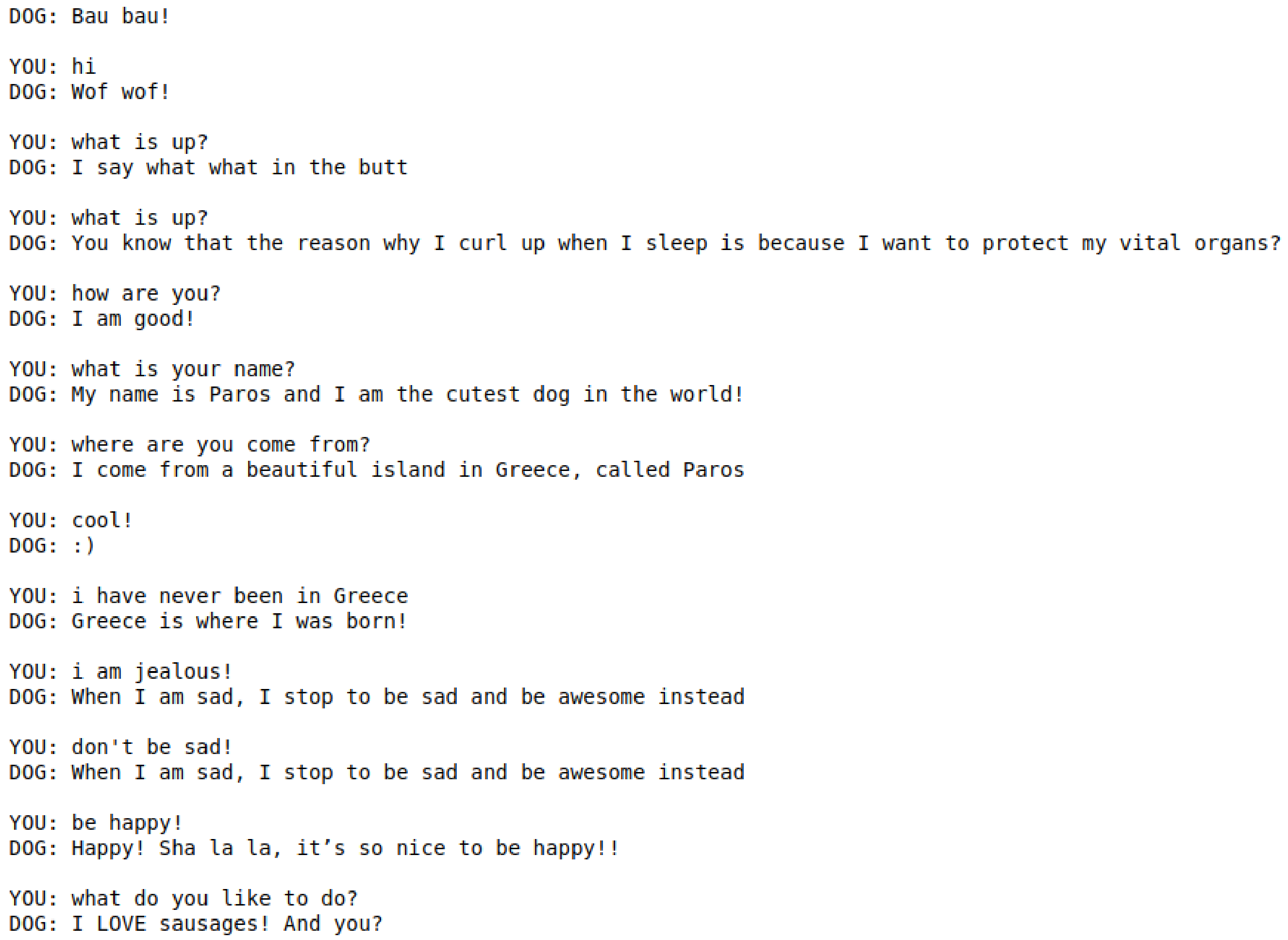
```
DOG: Bau bau!

YOU: what kind of dog is that?
DOG: Upload a picture!
YOU: paros.jpg
DOG: This looks to me like a crossbreed between a cocker spaniel (65.83%) and a Siberian husky (34.1
```

Although it is really unlikely that a cocker spaniel crossed with a husky, still the function gives a good hint (in case of the cocker spaniel) to understand what kind of breed is actually my dog.

To conclude, the chatbot to be fairly functional would need much more answers provided by a human. Nevertheless, looking around on mobile apps and website servers, this chatbot is already fair enough to compete with them. Providing an example, we take as a baseline another chatbot dog-character like: https://rebot.me/your-dog and we test a conversation of 12 turns first on my work and then we try to replicate similar questions there.

N.B.!! The conversation has been effectuated by a person who doesn't know how my chatbot or the other chatbot works:

My chatbot:



```
DOG: Bau bau!

YOU: hi
DOG: Wof wof!

YOU: what is up?
DOG: I say what what in the butt

YOU: what is up?
DOG: You know that the reason why I curl up when I sleep is because I want to protect my vital organs?

YOU: how are you?
DOG: I am good!

YOU: what is your name?
DOG: My name is Paros and I am the cutest dog in the world!

YOU: where are you come from?
DOG: I come from a beautiful island in Greece, called Paros

YOU: cool!
DOG: :)

YOU: i have never been in Greece
DOG: Greece is where I was born!

YOU: i am jealous!
DOG: When I am sad, I stop to be sad and be awesome instead

YOU: don't be sad!
DOG: When I am sad, I stop to be sad and be awesome instead

YOU: be happy!
DOG: Happy! Sha la la, it’s so nice to be happy!!

YOU: what do you like to do?
DOG: I LOVE sausages! And you?
```

We can see that overall the results is quite good! To notice, question “what is up” was not in the vocabulary so then when it was aked again the program retrieved another answer. The questions have been quite easy, most of them where already in the dataset so that the answers were safe. However, even when they were not, like in “I have never been in Greece” or “I am jealous” the program was able to provide a response partially convincing. However, a same answer was returned to two different queries “When I am sad, I stop to be sad and be awesome instead”. Now, manually updating the answers for these queries that have been automatically registred in the dataset will guarantee that in future conversations for this specific kind of phrases this won't happen again.

Now, I personally tested the same conversation with the baseline:

Your Dog: Oh, it's you again...

hi

Your Dog: You didn't feed me today so I went outside and found a nice rat. Look down.

what is up?

Your Dog: So, where is my bone?

what is up?

Your Dog: Whatcha up to?

how are you?

Your Dog: Good, how are you?

i am good

Your Dog: nice :)

what is your name?

Your Dog: Anonymouse

where are you come from?

Your Dog: Scratch my ear

cool!

Your Dog: I know right?!

I have never been in Greece

Your Dog: ?

I am jealous!

Your Dog: Whatcha up to?

Don't be sad!

Your Dog: Sup you

i am great

Your Dog: !!no

what do you like to do?

Your Dog: I am going to say a bad word........POOP! That felt good to let out.

To notice is that even the chatbot also doesn't reply with the same answer twice to the same question as we can see with the question "What is up?". For the rest, it seems that it has less topics covered than mine, even though those were really basic questions (but I ingore the system how it has been implemented). When it doesn't know an answer, it seems that it simply randomly some templated phrases.

In the future, the conversational part of my chatbot could be updated with more asnwers, implemented with more answer options for a same query, fine-tuned with some keywords matching patterns that work on a confidence level.

BONUS:

I am a crossbreed between a chiuaha and a husky!