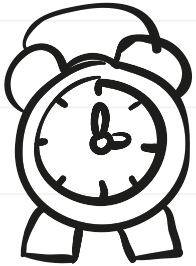
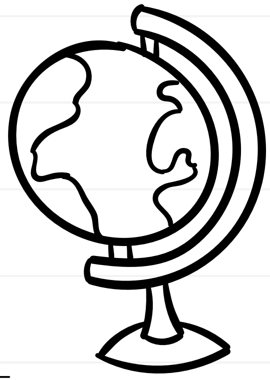


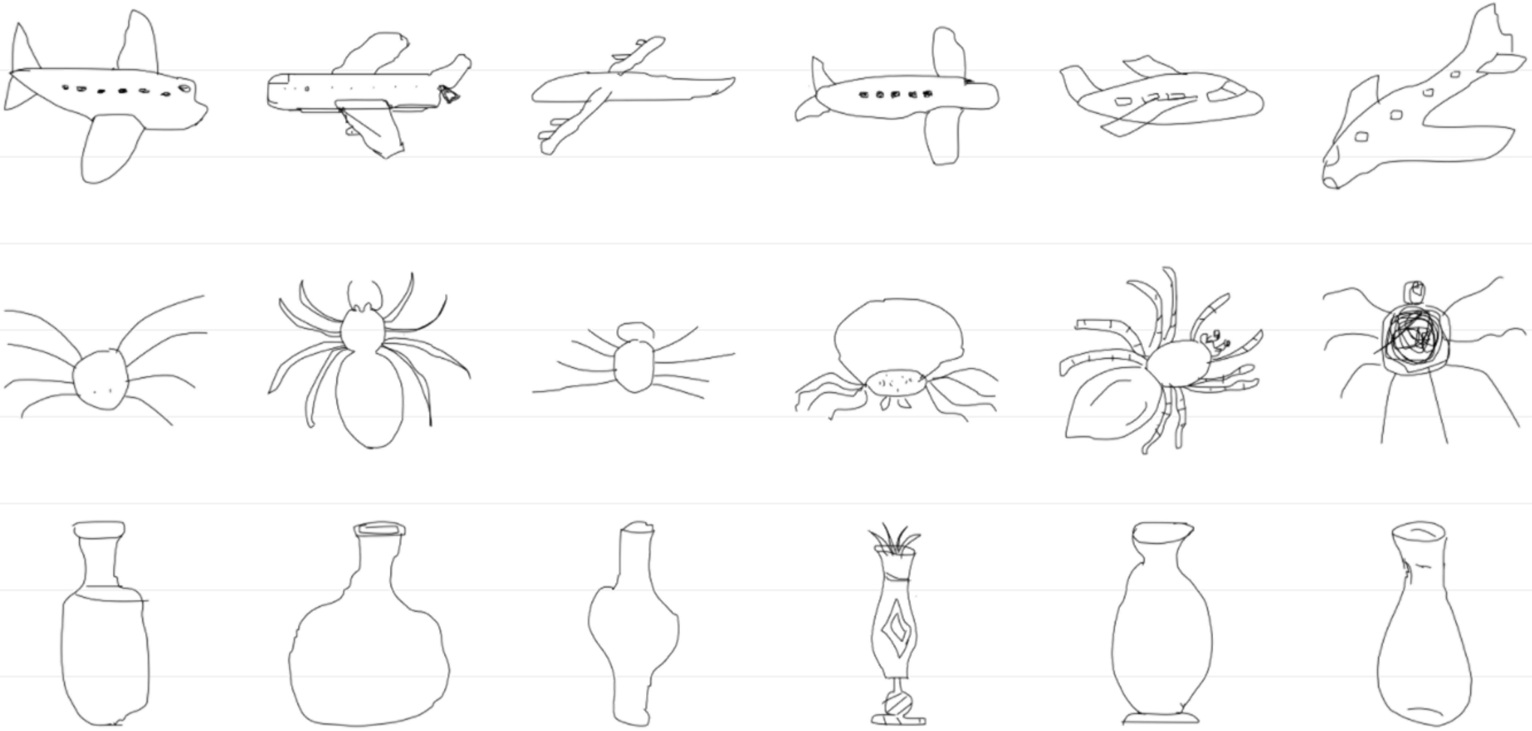
SKETCHES RECOGNITION



— Students: F. Minutella | A Schiavo | M. Daole | M. Gómez || Professors: C. Gennaro | F. Falchi | G. Amato | P. Bolettieri —

PROJECT DESCRIPTION

1. Implement index LSH to allow fast similarity search on deep features and create an image search engine on top of it
2. Use the pre trained Deep Neural Network Inception to extract features from the dataset Sketches and the distractor MirFlickr
3. Index the extracted features using your search engine
4. Measure the retrieval performance of the image search engine
5. Fine tune the Inception for Sketches
6. Use the fine tuned Inception to extract features from Sketches and MirFlickr
7. Index the new extracted features using your search engine
8. Measure the retrieval performance of the image search engine using the new features
9. Compare the performance of the two features
10. Optional:
    - Build a web based user interface for your web search engine

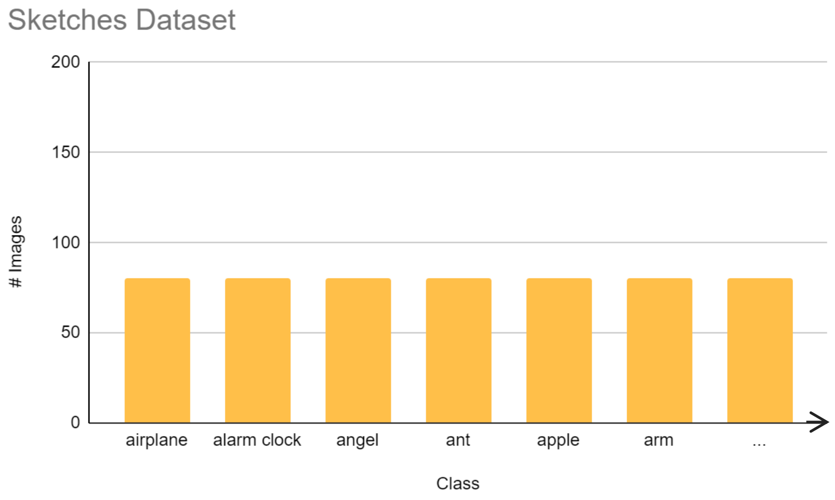


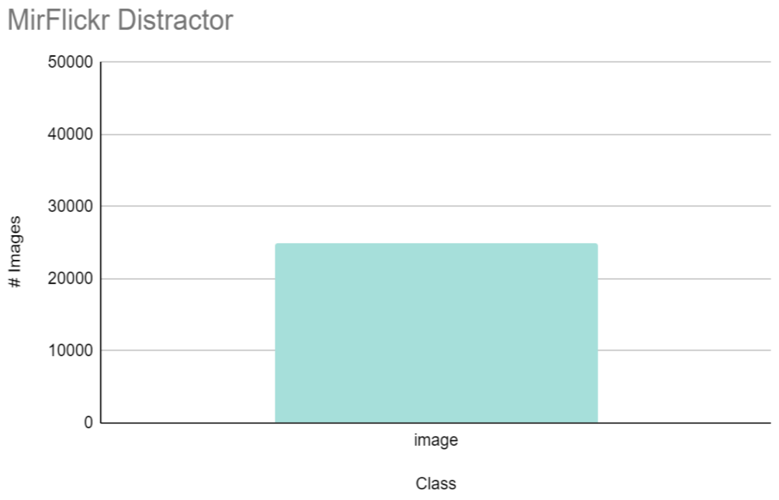
Our goal is to train machines to recognize and generalize abstract concepts in a manner similar to humans. As a first step towards this goal, we train our model on a dataset of hand-drawn sketches.

Distribution of our Datasets

250 Class and 20K Images

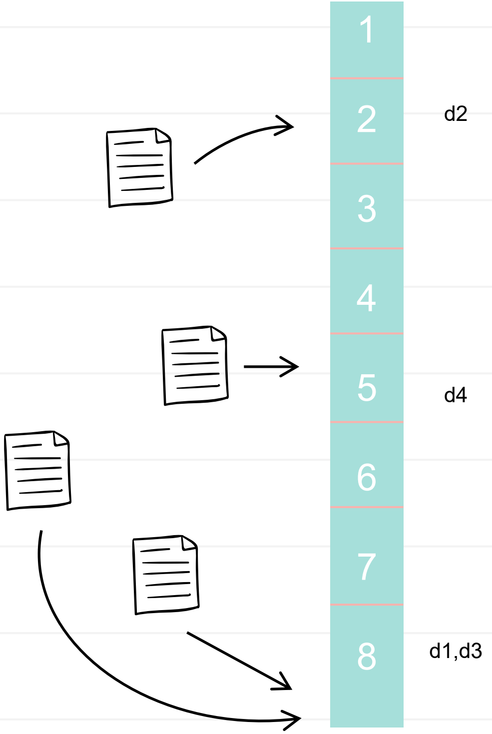
1 Class and 25K Images





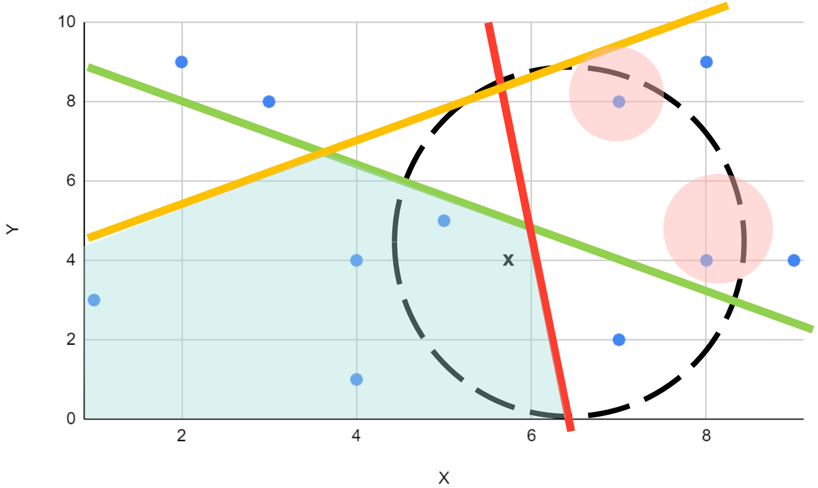
LSH - Locality Sensitive Hashing

- We don't necessarily insist on the exact answer; instead, determining an approximate answer should suffice.
    - Similar Documents => Similar Hash-Code
- For each document d:
    - Generate K-bit hash-code
    - Insert Document into hash-table
    - Collision => possible duplicate
        - Compare to documents in same bucket



LSH - Locality Sensitive Hashing

- Hash functions g() such that given any two objects p,q
  - If d(p,q) > c*r then Pr[g(p)= g(q)] is small
  - If d(p,q) ≤ r then Pr[g(p) = g(q)] is not so small



Definition: $g(p)=<h_1(p), h_2(p), ..., h_k(p),>$

Where: $h_i(p) = \lfloor (p*X_i+b_i)/w) \rfloor$

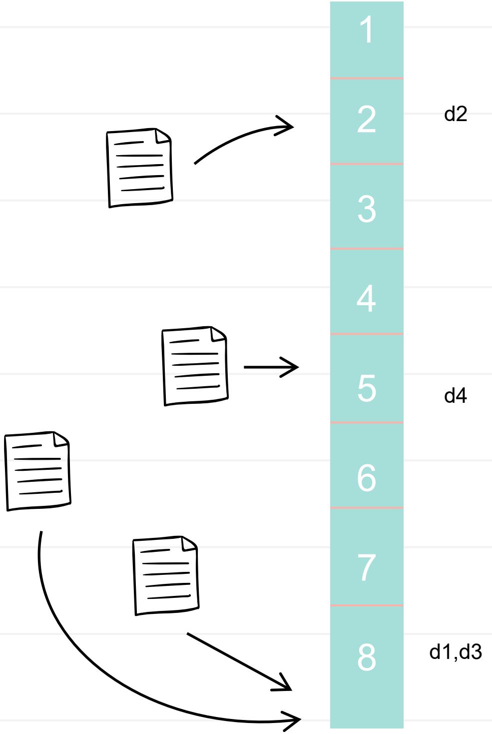
We compare x only to training points in the same region R

Inexact: missed neighbors

- Repeat with different $h_1, h_2, ..., h_k$

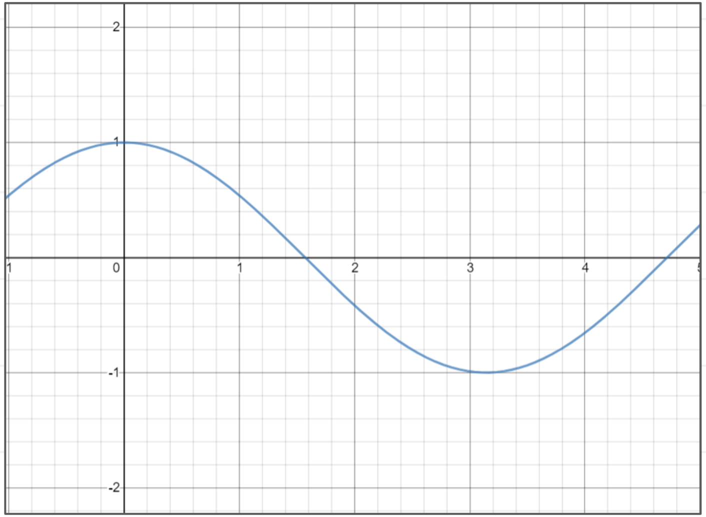
LSH – Implementation

- Preprocessing: Hash function selecting $\langle g_1, g_2, ..., g_L \rangle$
- Insertion: Any point p Insert into L buckets $\langle g_1(p), g_2(p), ..., g_L(p) \rangle$
- Query execution: With the query q retrieve all point from L buckets $g_1(q)$ and reorder according to the original distance function

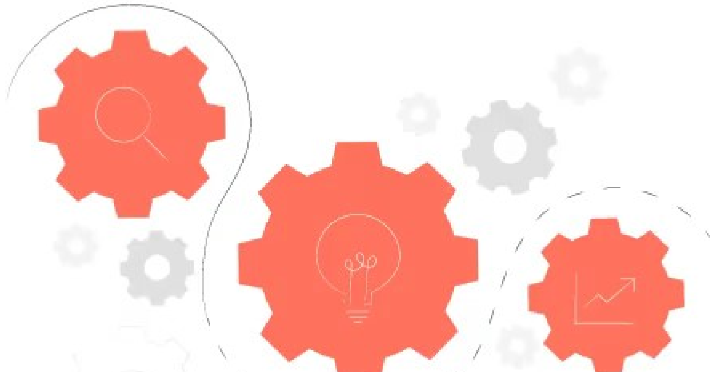


LSH – Bitwise

- The random projection method of LSH called SimHash is designed to approximate the cosine distance between vectors.
- The basic idea of this technique is to choose a random hyperplane at the outset and use the hyperplane to hash input vectors.
    - $h(v) = \pm 1$
- We called Bitwise because, instead ± 1 we convert this approach to 0 or 1 (bit)
- This approach is faster and require less storage



No Index Structure – Test



- Inside the project, we create an index with the same function structure of the LSH Index for test comparations.

- Why we do that ?
  - Our idea was use the same structure of our LSH index, without use the computational complexity of an index. This, for search without index.
  - We create a set of test with an exact index (No Index Structure) and after compare the results with an LSH index

Features Extraction

- Feature Extraction using the pretrained convolutional base

01

Create a image normalization preprocess with our images

02

Loading the pretrained DNN

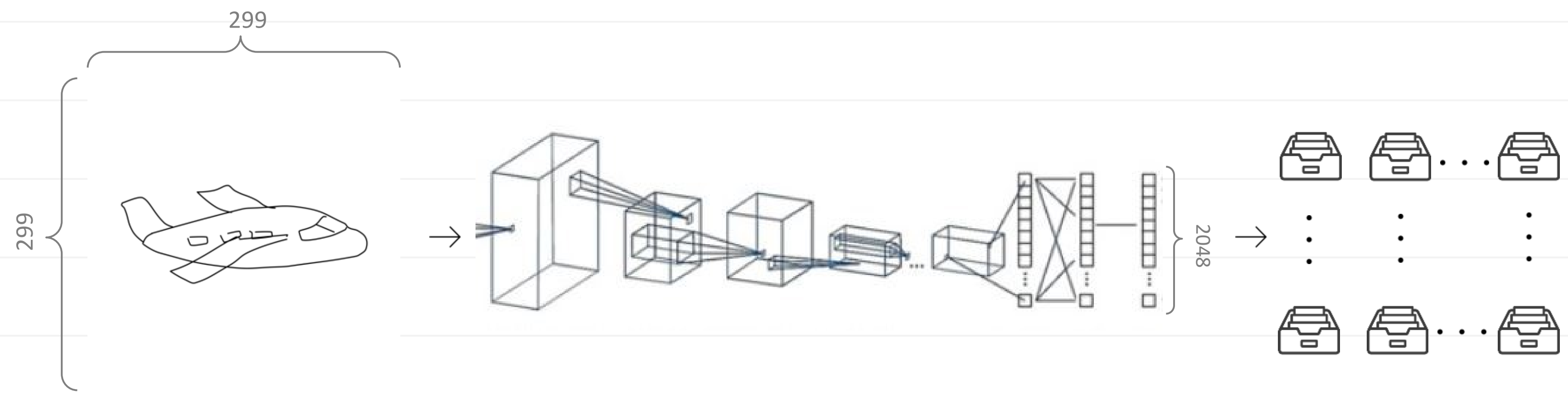
03

Extract of the features

04

The result is our feature vector

High Level Architecture



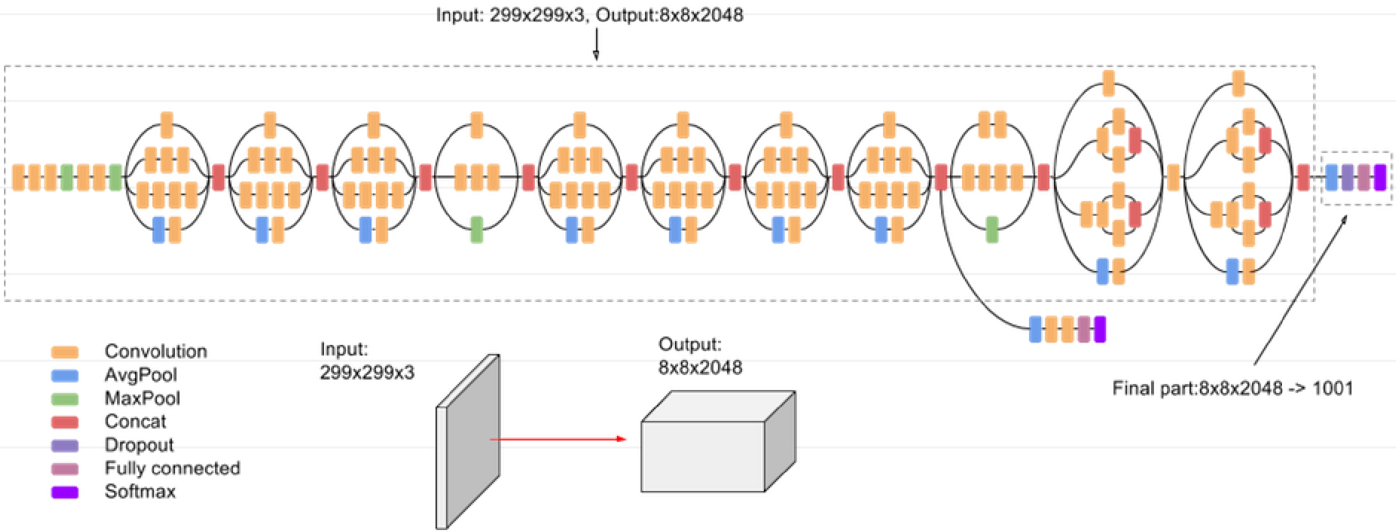
Input Image

Features Extraction

LSH Index

Inception V3

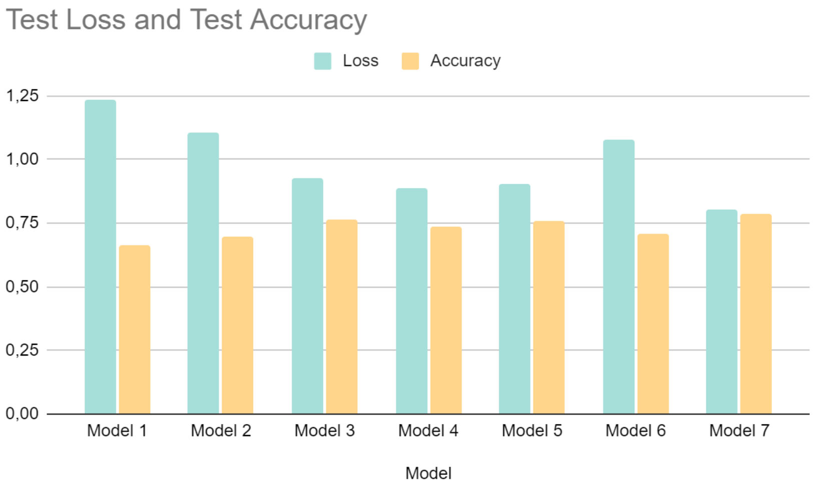
- Have a very important milestone in the development of CNN classifiers.
- The fundamental idea behind the Inception Neural Network is the inception block
- Intermediate Classifiers to solve Vanishing Gradient.



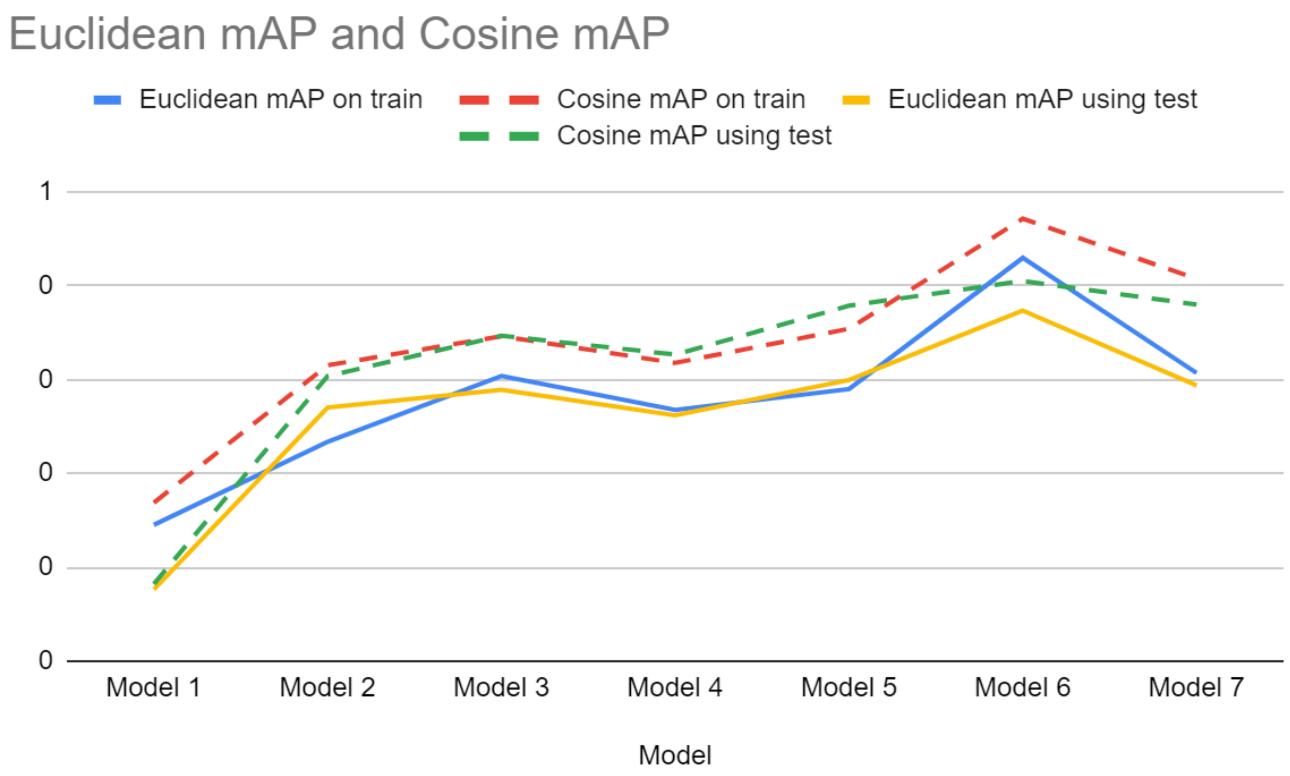
Model Test and Performance

Model name	Train		Test		Euclidean mAP on train	Cosine mAP on train	Euclidean mAP using test	Cosine mAP using test
	Loss	Accuracy	Loss	Accuracy				
Model 1	1.2724	0.6617	1.2328	0.6648	0.1452	0.1687	0.0763	0.0821
Model 2	1.1013	0.7005	1.1035	0.6986	0.2336	0.3150	0.2701	0.3035
Model 3	0.9033	0.7612	0.9272	0.7620	0.3035	0.3460	0.2889	0.3464
Model 4 (Siamese)	0.4611	0.8157	0.8868	0.7373	0.2674	0.3175	0.2618	0.3264
Model 5	0.8940	0.7667	0.9065	0.7610	0.2896	0.3539	0.2993	0.3784
Model 6	0.6899	0.8087	1.0764	0.7090	0.4294	0.4711	0.3731	0.4047
Model 7	0.7600	0.7867	0.8022	0.7840	0.3068	0.4071	0.2935	0.3797

Model Test and Performance

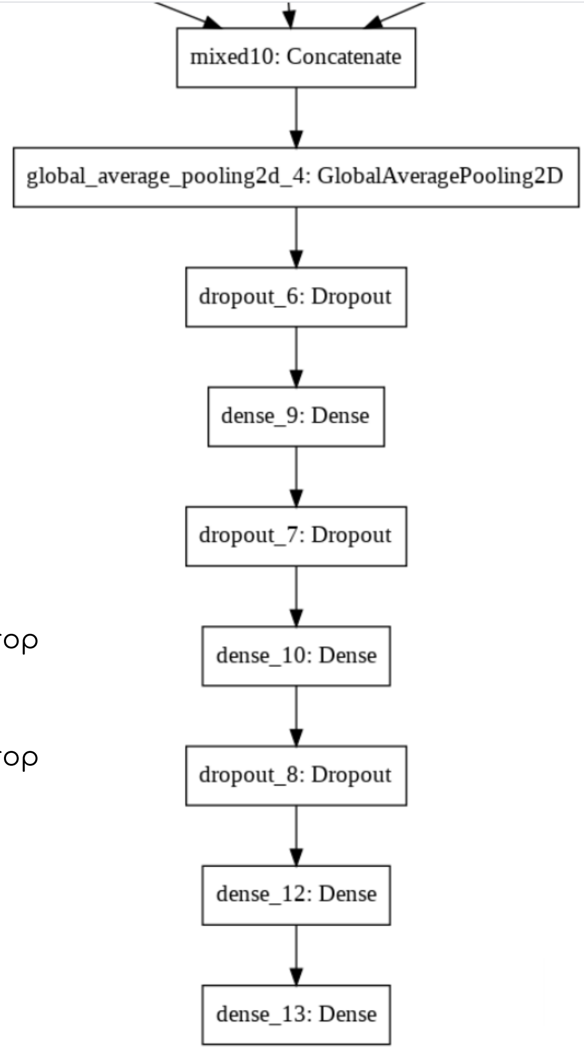


Model Test and Performance



Best Model Chosen: Model 6

- We choose the model with the next custom layers:
    1. Pre-trained model Inception V3
    2. Global spatial average pooling layer
    3. First droupout layer with a fraction of 0.5 input units to drop
    4. First fully connected layer with 2048 dimensionality
    5. Second droupout layer with a fraction of the 0.5 input units to drop
    6. Second fully connected layer with 2048 dimensionality
    7. Second droupout layer with a fraction of the 0.5 input units to drop
    8. Third fully connected layer with 2048 dimensionality
    9. Fourth fully connected layer with 250 dimensionality



LSH Tuning (G and H)

LSH	Test mAP		Test IE		Bucket Purity AVG	Bucket Purity STD	# Buckets	# Items	# AVG per BUCKET	# STD per BUCKET
	Euc	Cos	Euc	Cos						
G = 3, H = 3	0,05	0,06	355	297	0,72	0,32	4885	120000	24,5	183,97
G = 3, H = 5	0,01	0,02	3441	3485	0,87	0,24	24363	120000	4,93	36,3
G = 4, H = 2	0,17	0,20	11,2	14,3	0,62	0,34	1188	160000	134,68	677,15
G = 5, H = 1	0,31	0,37	0,52	0,46	0,52	0,35	150	200000	1333,3	3571,97
G = 5, H = 2	0,19	0,22	10	11,81	0,61	0,35	1629	200000	122,77	642,32
G = 7, H = 2	0,24	0,26	5,8	3,9	0,62	0,34	2319	280000	120,7	722,83
G = 8, H = 2	0,25	0,27	1,9	2,69	0,62	0,35	2548	320000	125,59	688,5

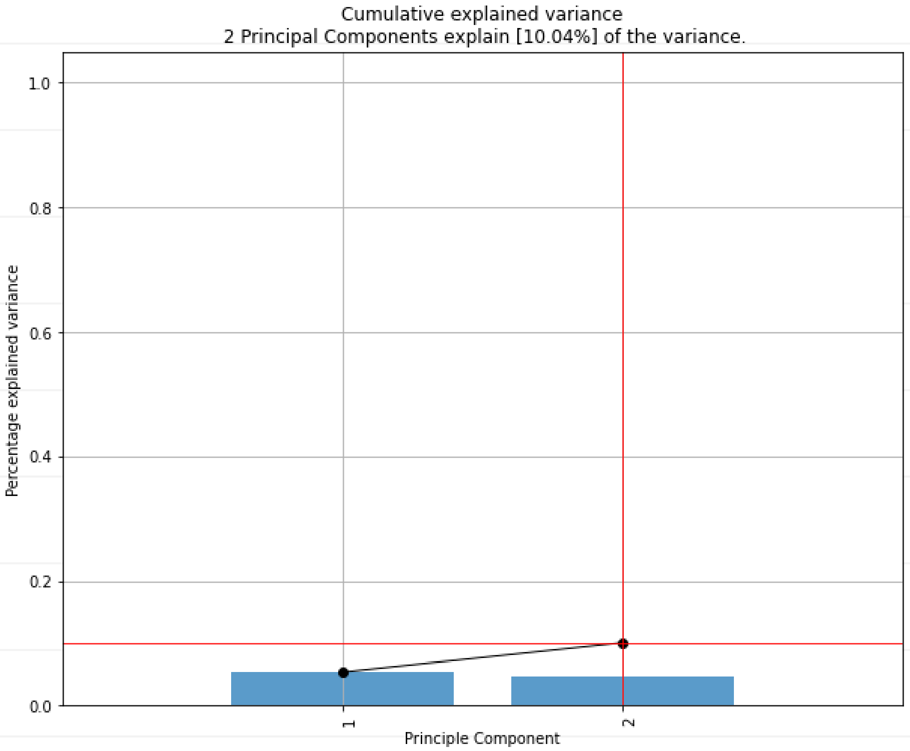
LSH BitWise Tuning (G and H)

LSH BitWise	Test mAP		Test IE		Bucket Purity AVG	Bucket Purity STD	# Buckets	# Items	# AVG per BUCKET	# STD per BUCKET
	Euc	Cos	Euc	Cos						
G = 3, H = 1	0,35	0,42	0,26	0,26	0,52	0,27	6	120000	20000	8807
G = 3, H = 5	0,30	0,33	5	4,67	0,26	0,27	96	120000	1250	2733
G = 3, H = 6	0,28	0,32	8	8,2	0,3	0,25	192	120000	625	1502,5
G = 4, H = 4	0,32	0,39	1,36	1,37	0,37	0,32	64	160000	2500	3220,32
G = 5, H = 2	0,36	0,42	0,31	0,31	0,35	0,36	20	200000	10000	8618,35
G = 5, H = 5	0,31	0,39	2,23	2,2	0,29	0,26	160	200000	1250	2347

PCA - Principal Component Analysis

- Is a standart techniche in the field of signal processing for data compression.

- Is a linear dimensionality reduction using Singular Value Decomposition of the data to project it to a lower dimensional space.
- We try it !



ACCURACY AND MAP ANALYSIS CLASS BY CLASS

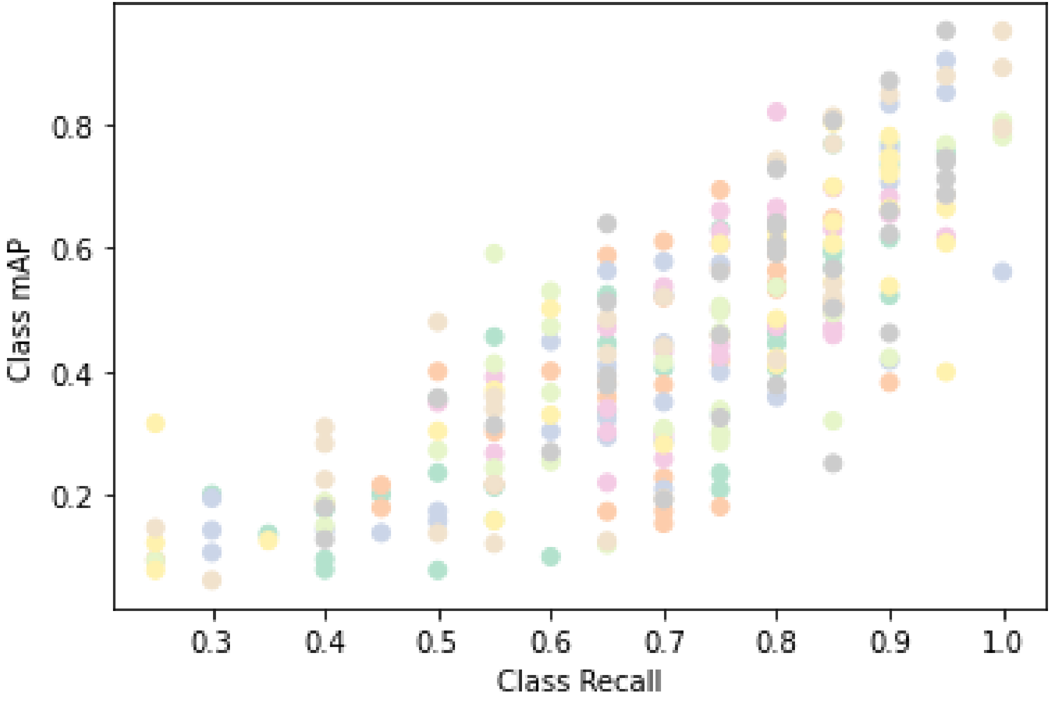
01 We create a set of query test class by class

02 For all the previous queries, we calculate the AP

03 mAP, min(AP), max(AP), std(AP), range(AP)

04 Return the Worst Sketch result and Best Sketch result

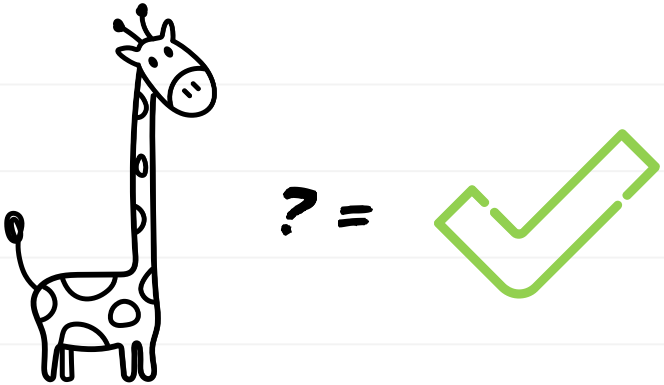
Recall Vs Mean Average Precision



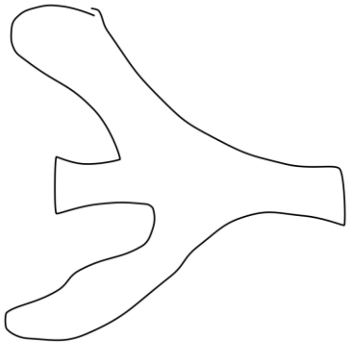
Correlation coefficient:

0.7863

BEST MAP AND ACCURACY RESULTS

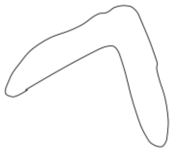


Class Boomerang: mAP 0.77 Recall 0.90



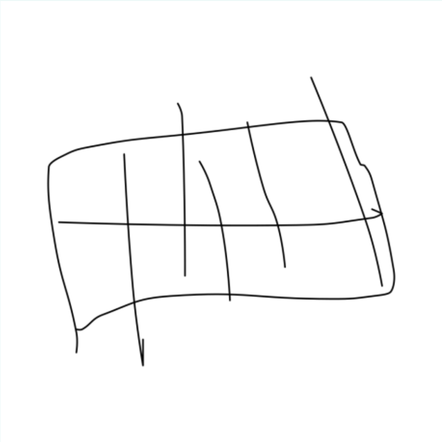
Worst sketch:
AP = 0.0125

Best sketch:
AP = 0.936

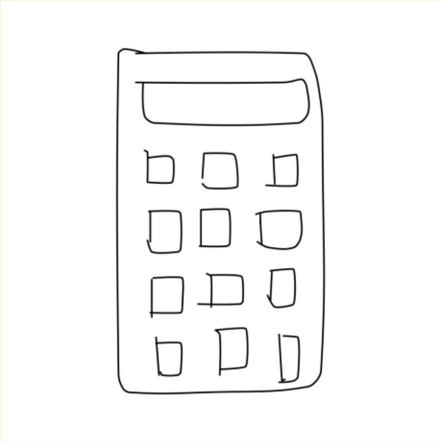


Some other sketches from the boomerang class:

Class Calculator: mAP 0.65 Recall 0.85



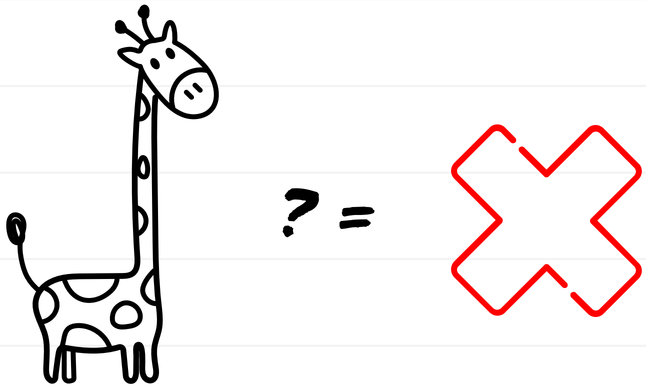
Worst sketch:
AP = 0.0131

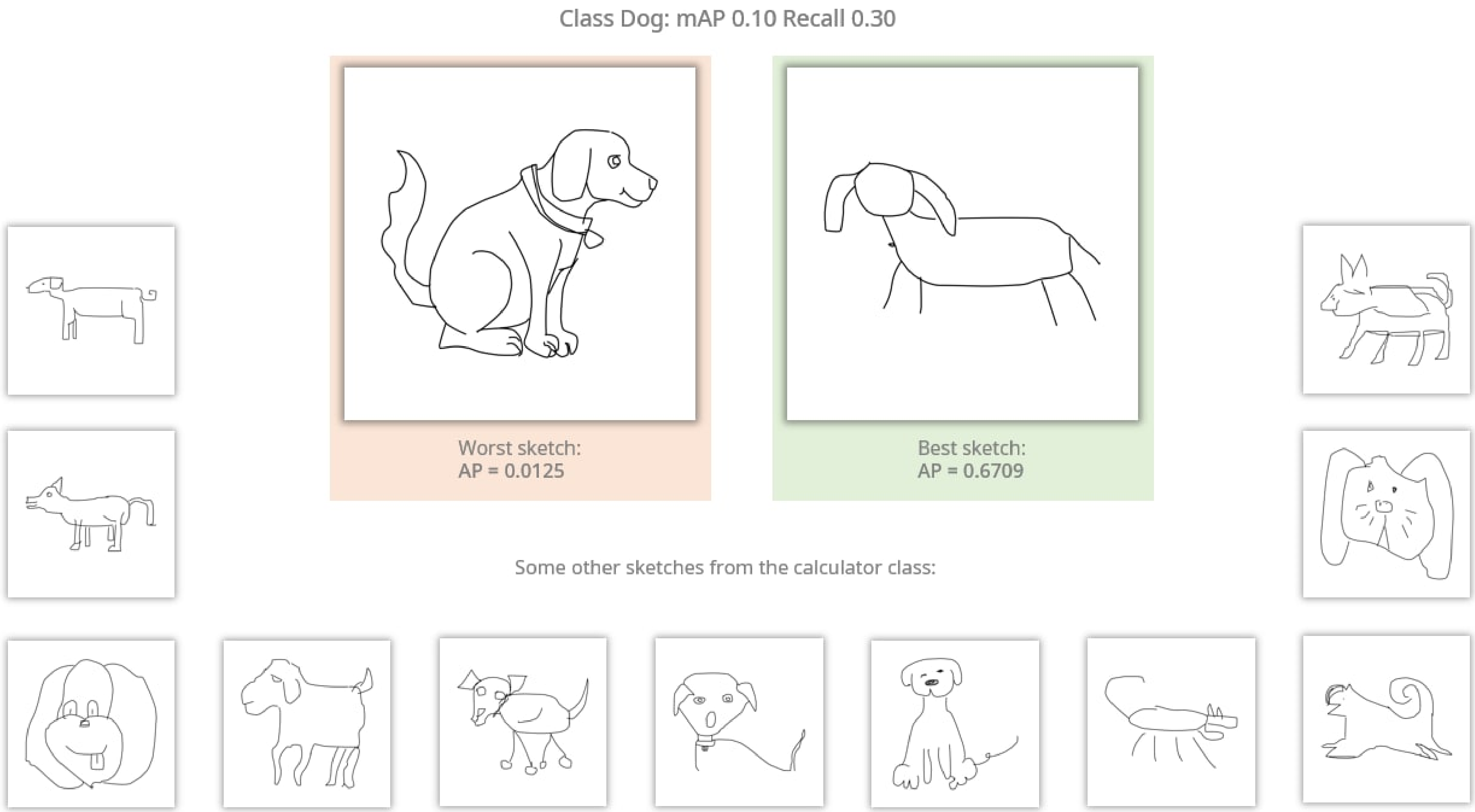


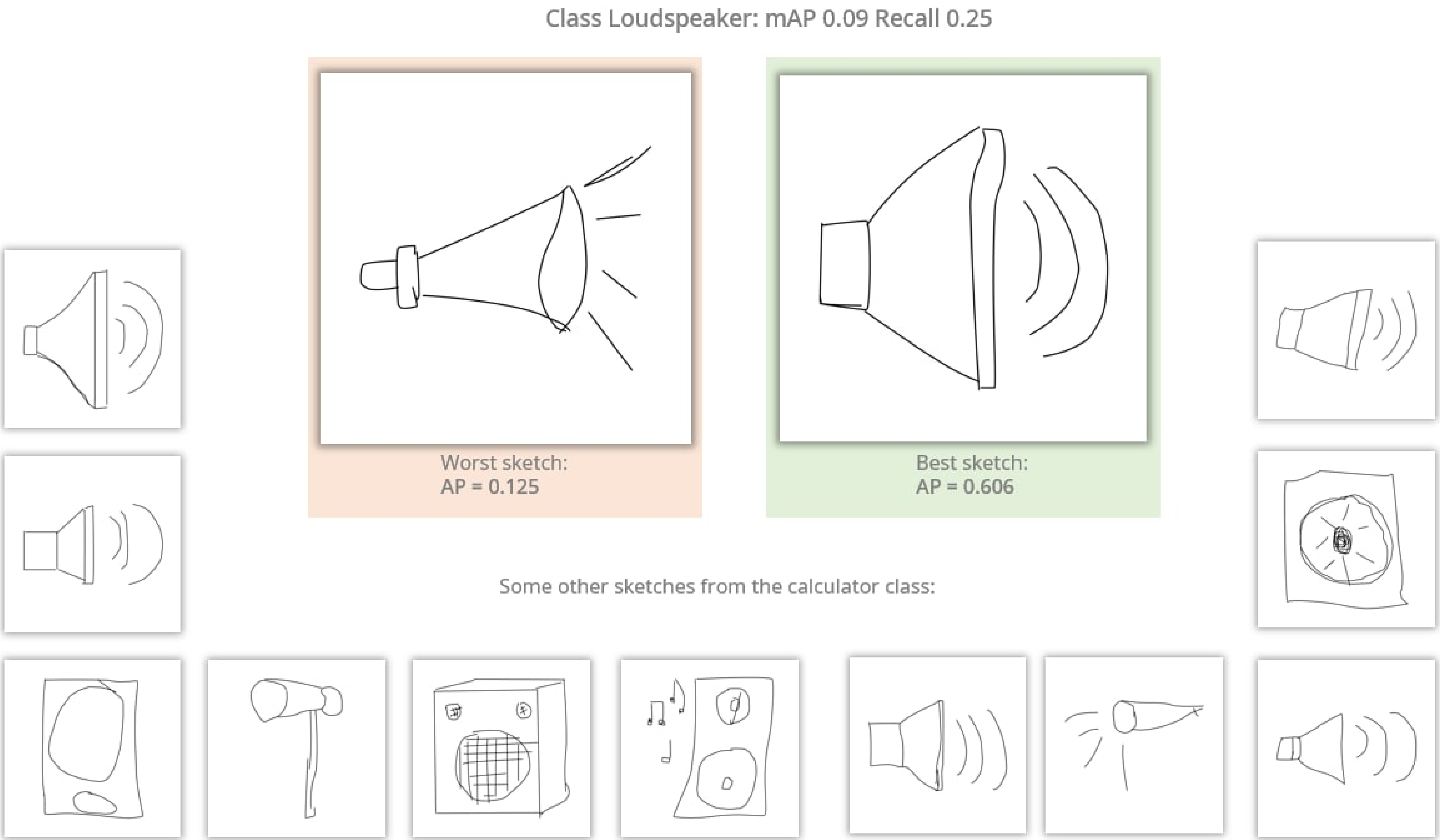
Best sketch:
AP = 0.8586

Some other sketches from the calculator class:

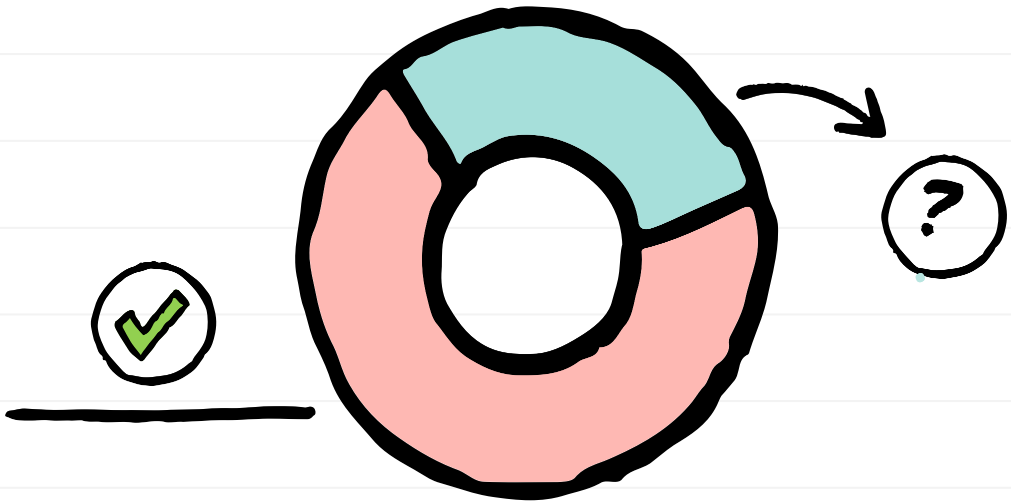
Worst MAP and Accuracy results




Class Dog: mAP 0.10 Recall 0.30
Worst sketch:
AP = 0.0125
Best sketch:
AP = 0.6709
Some other sketches from the calculator class:

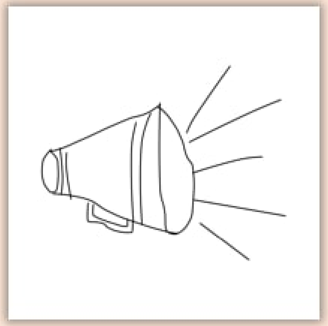
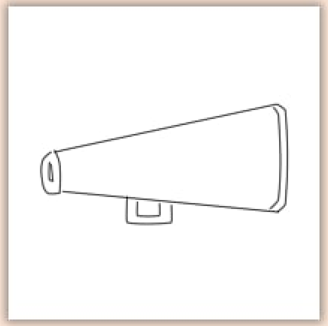
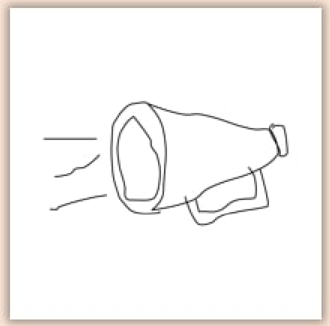
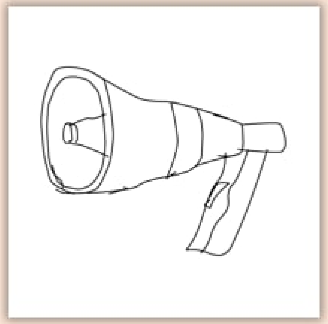

Class Loudspeaker: mAP 0.09 Recall 0.25
Worst sketch:
AP = 0.125
Best sketch:
AP = 0.606
Some other sketches from the calculator class:

Most Frequent Misclassifications

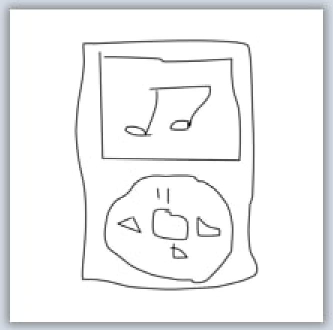
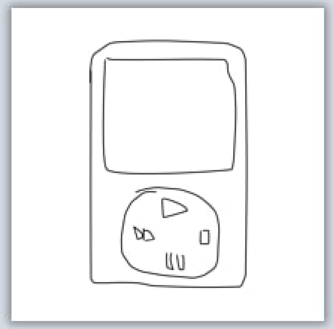
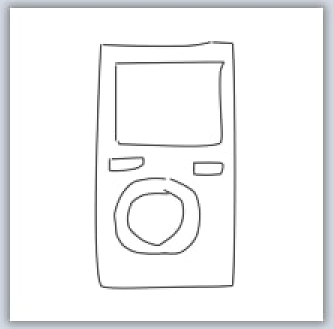


Loudspeaker Class: most frequent misclassifications

4/20 times misclassified as Megaphone:

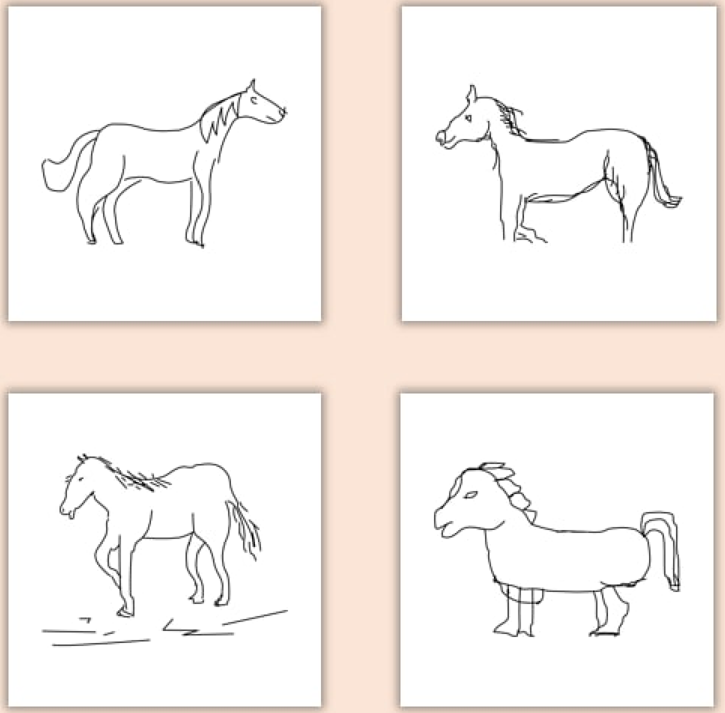





3/20 times misclassified as Ipod:

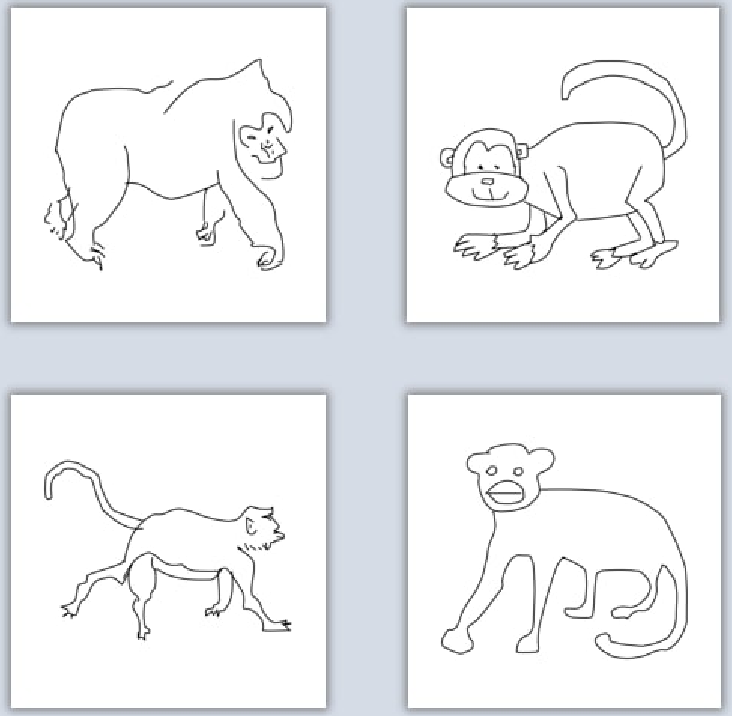





Dog Class: most frequent misclassifications

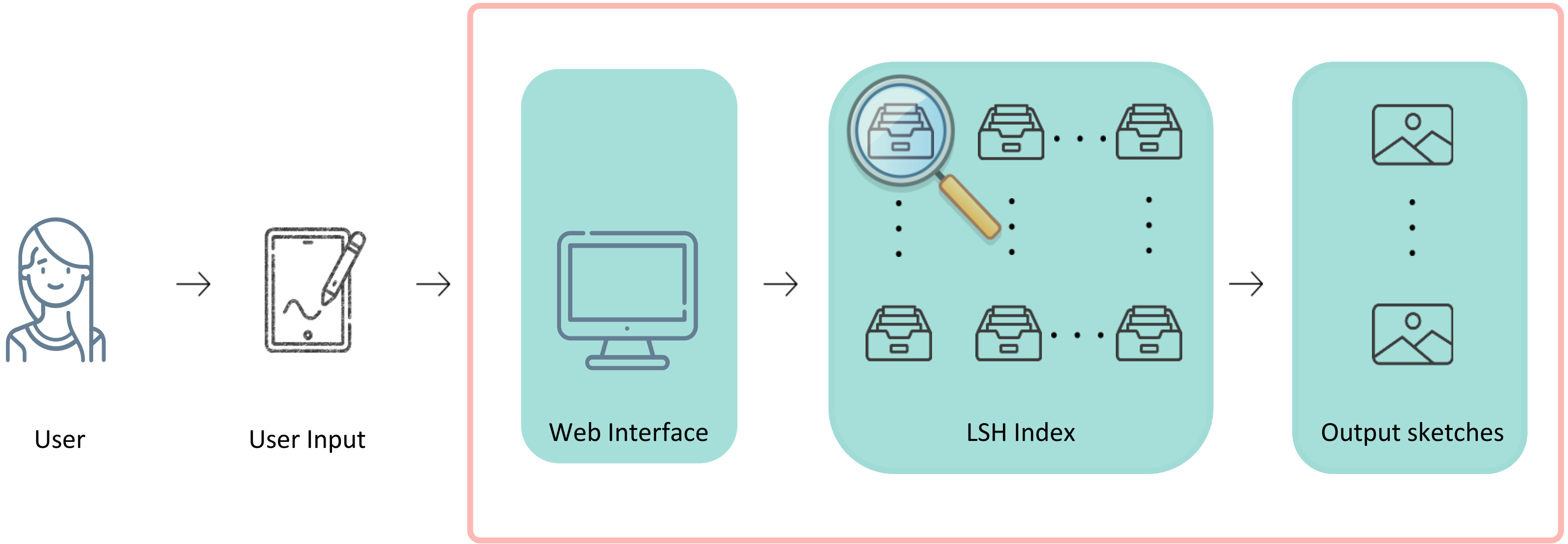
4/20 times misclassified as Horse:



3/20 times misclassified as Monkey:



WEB SKETCHES RECOGNITION ARCHITECTURE



DEMO

Project Source Code

- Analysis_By_Class
- MIRCV
- MIRCV_Functions
- Pca_Try
- accuracy_assessment
- evaluate_models
- lsh_finetuning
- lsh_finetuning_bucket_dispersion
- mAP_by_class
- save_index
- siamese_finetuning
- sketches
- web

RESOURCES

Papers

- Rethinking the Inception Architecture for Computer Visual
- A Neural Network for PCA and Beyond
- Theory and Applications of b-Bit Minwise Hasing

THANKS!