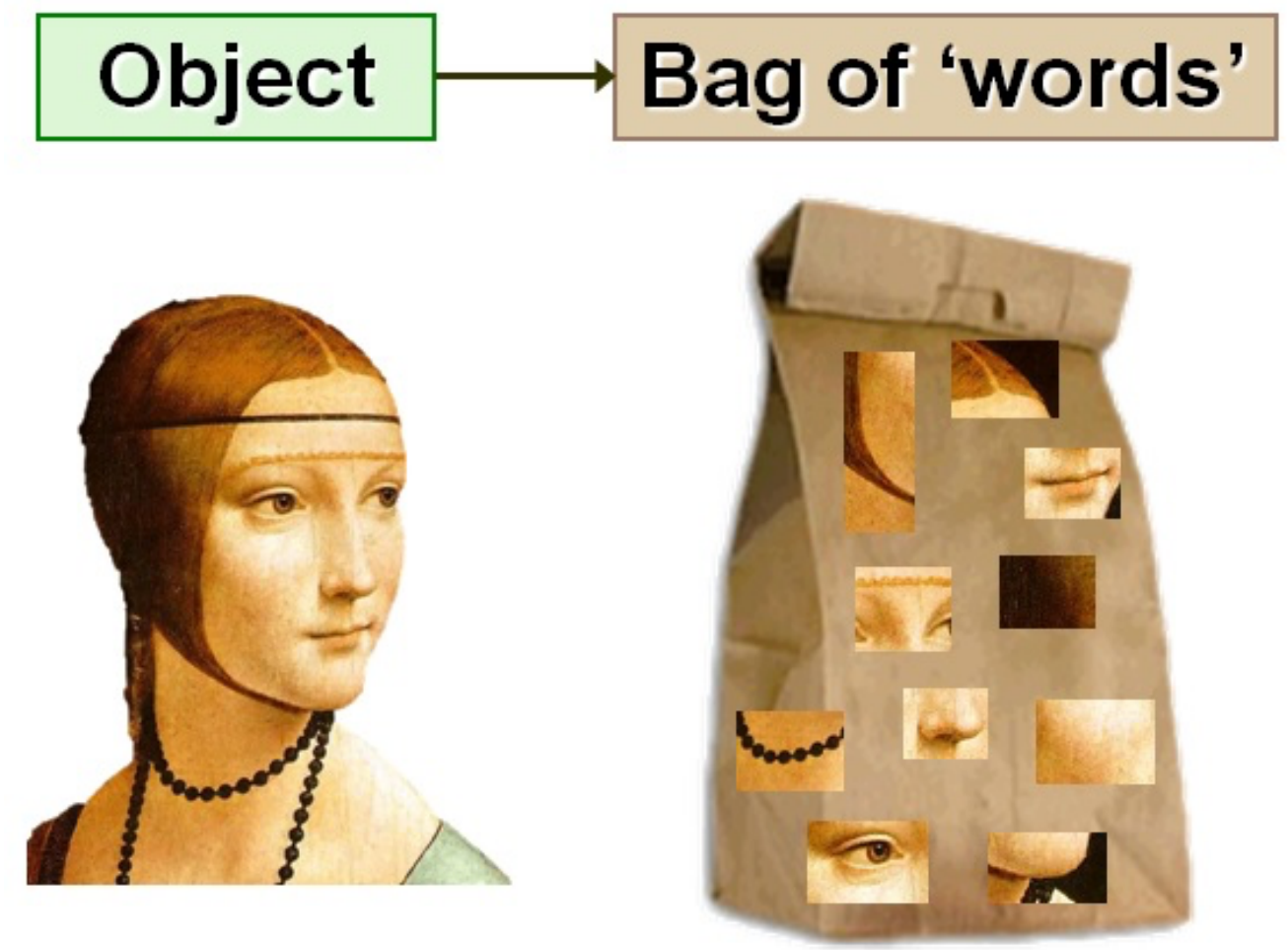
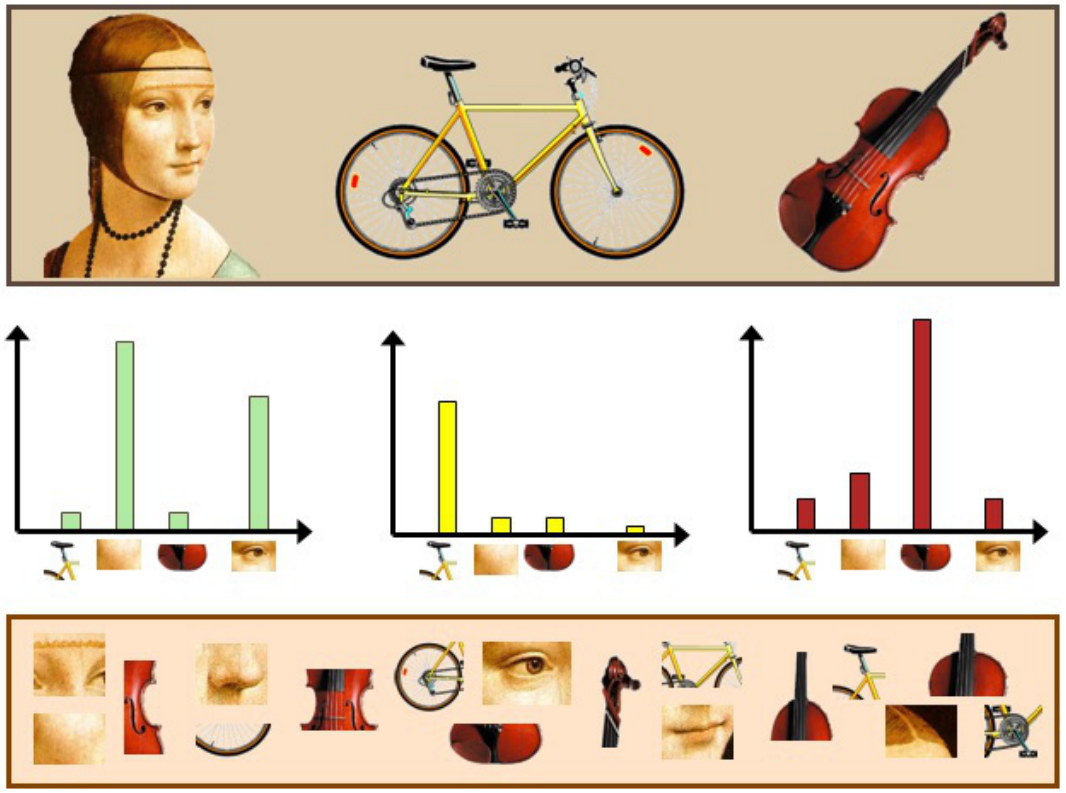
BoW Image Classification

adam.karwan@gmail.com



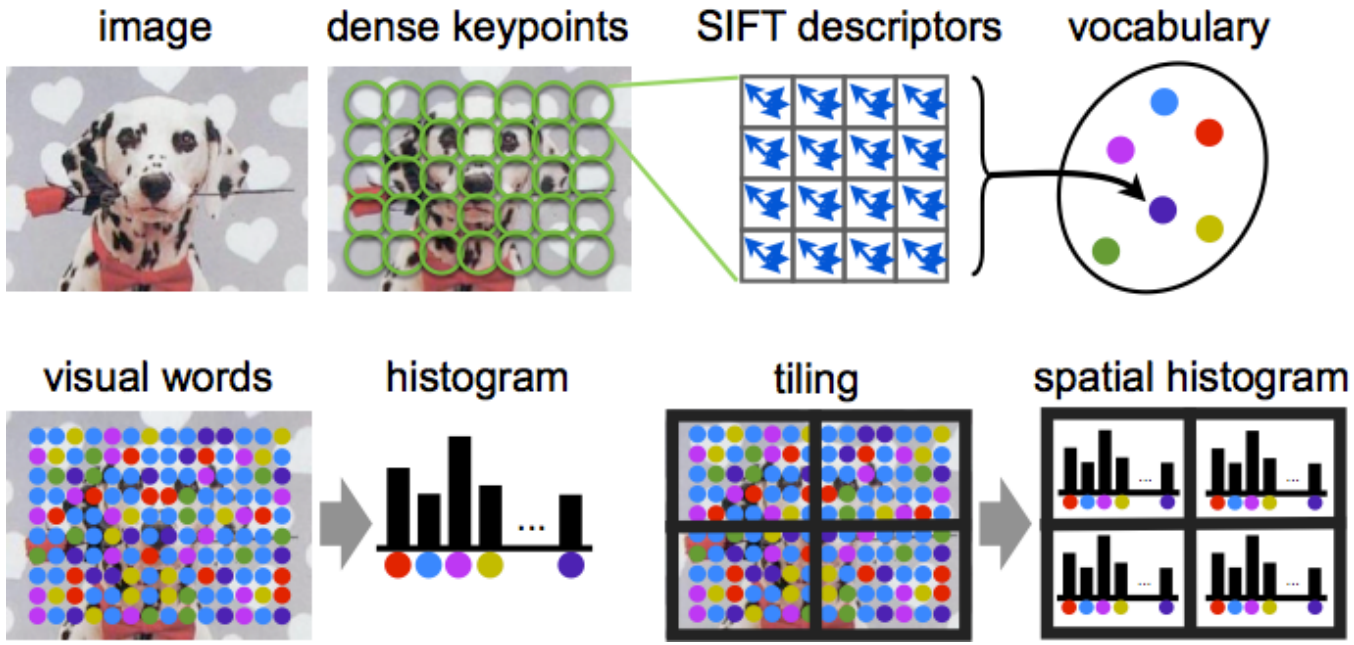
Scene Recognition with Bag of Words



www.cc.gatech.edu/classes/AY2016/cs4476_fall/results/proj4/html/vdedhia6/index.html

youtube.com/watch?v=HWIkcMakz-s **[1h] Bag of Visual Words (Cyrill Stachniss, 2020)**
youtube.com/watch?v=a4cFONdc6nc **[5min]**

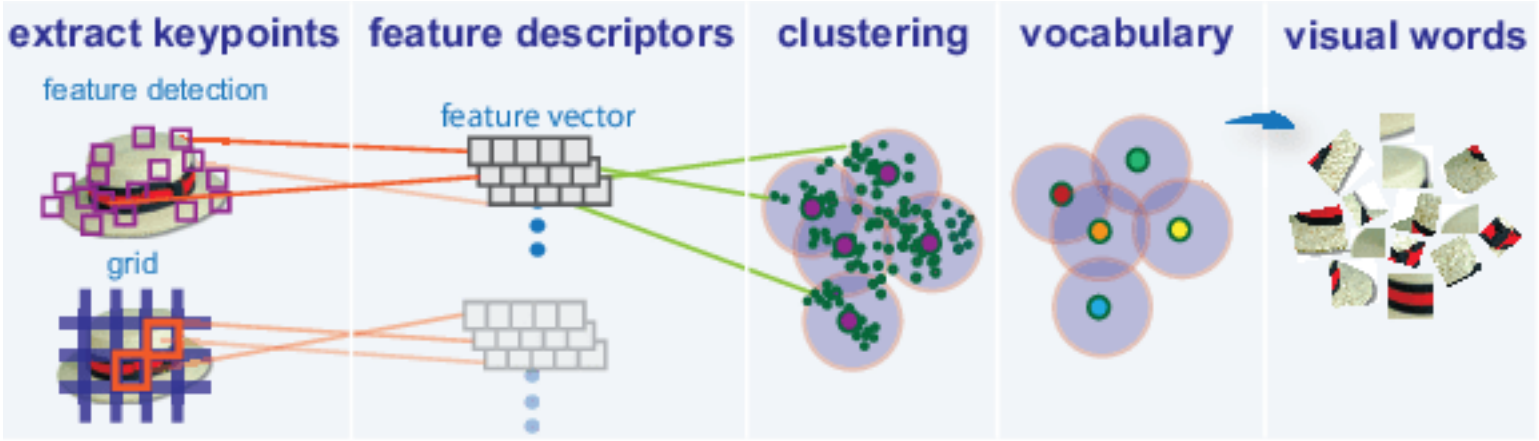
Starting Point ICCV 2005

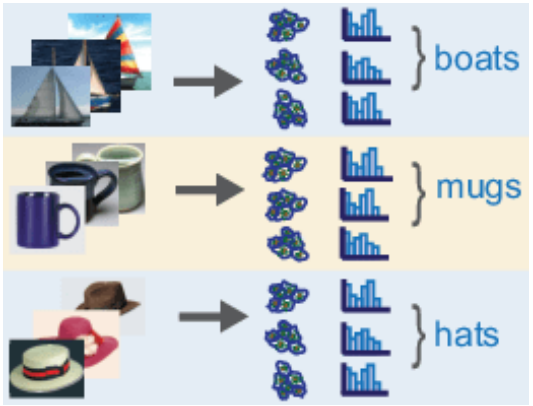


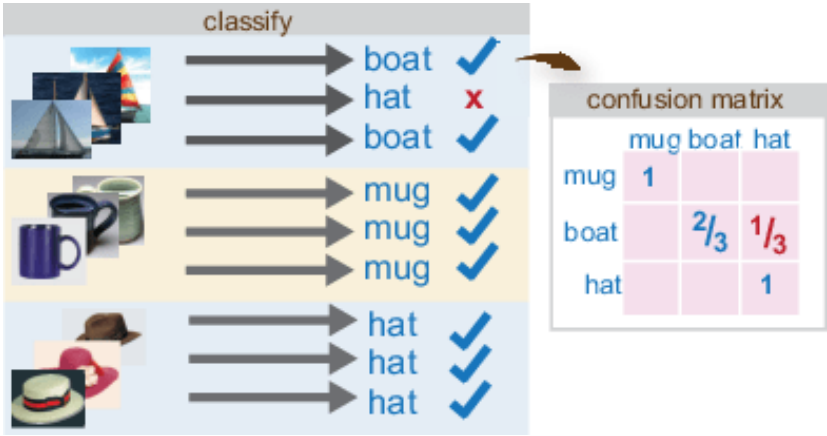
people.csail.mit.edu/fergus/iccv2005/bagwords.html

www.di.ens.fr/willow/events/cvml2011/materials/practical-classification

Mathworks - MATLAB

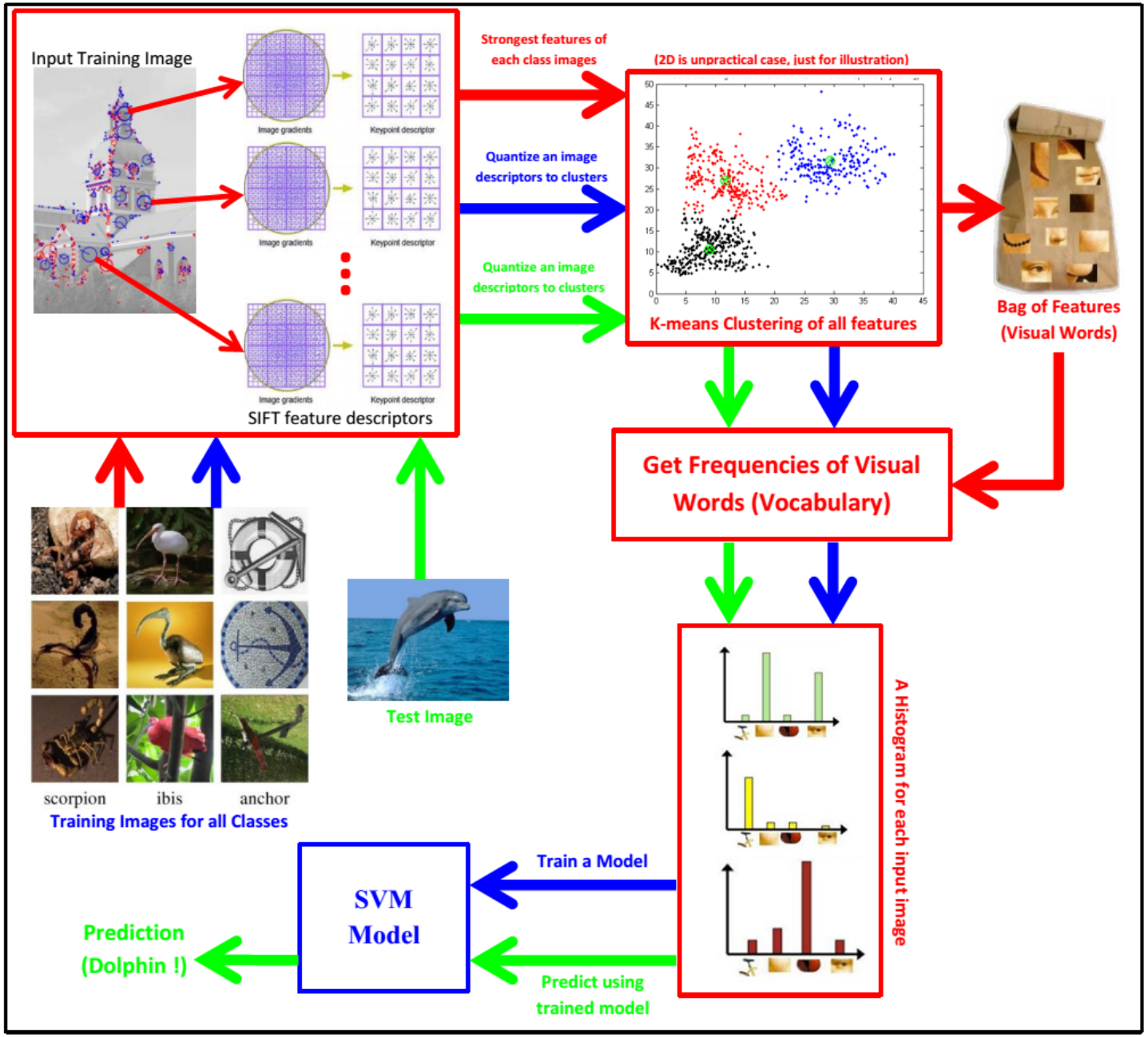






www.mathworks.com/help/vision/ug/image-classification-with-bag-of-visual-words.html

BoW + SVM



heraqi.blogspot.com/2017/03/BoW.html

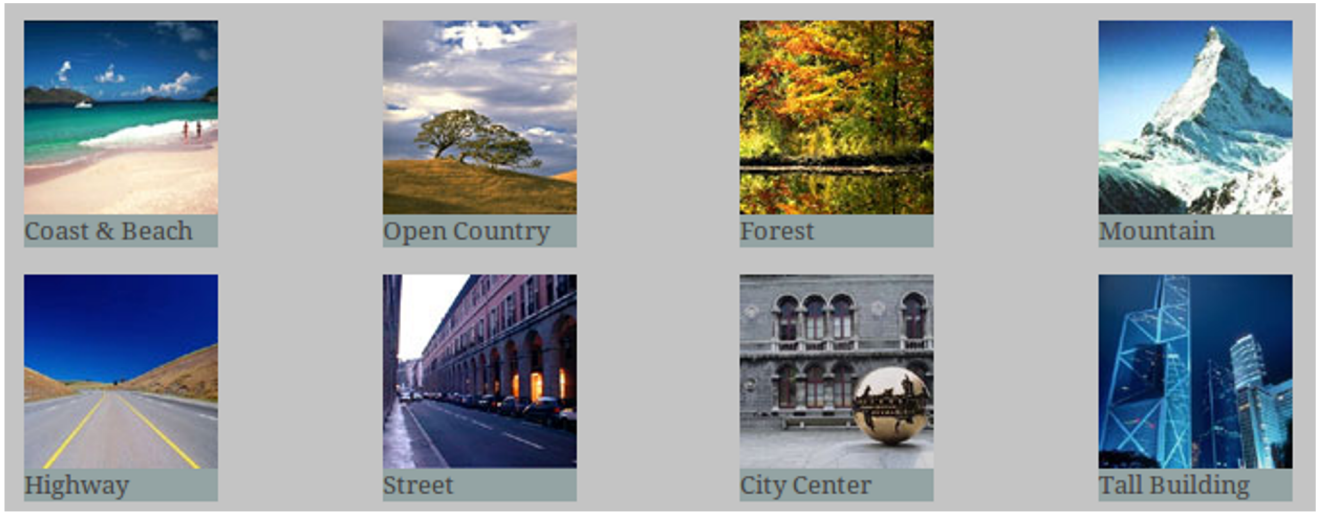
www.mathworks.com/matlabcentral/fileexchange/62217-image-classification-with-bag-of-visual-words

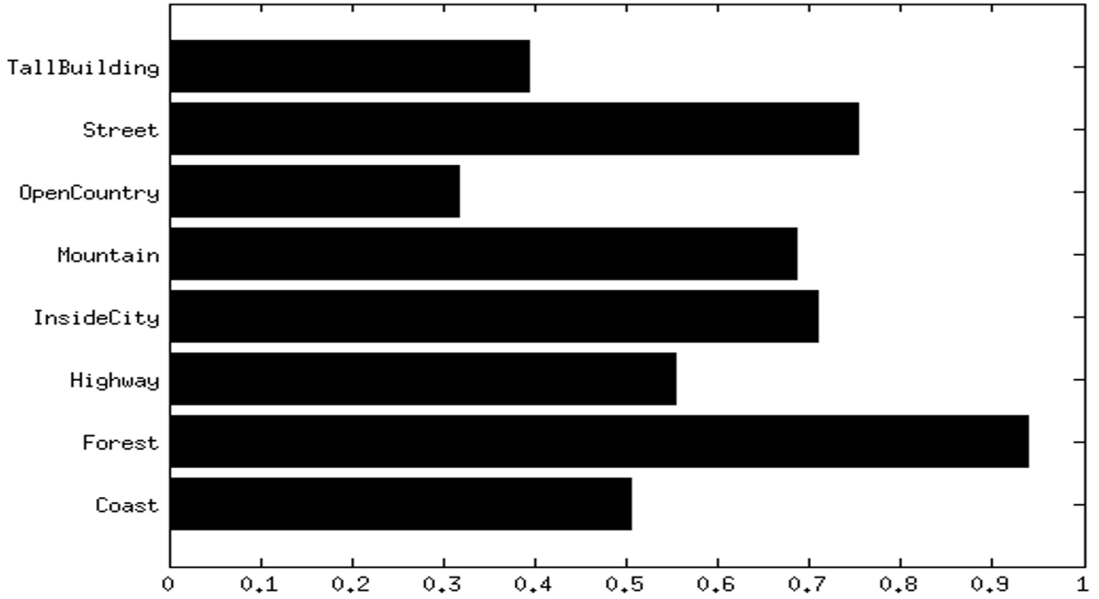
First Results, 2013 year, 8 categories

Dataset	cvcl.mit.edu/database.htm [2013]
All images	**2688**
Categories	**8**
Train / Test	0.5
Words	300
Topics	20
Overall accuracy	**59,45%**

places.csail.mit.edu/index.html
205 *scenes*, 2.5M+ *images*

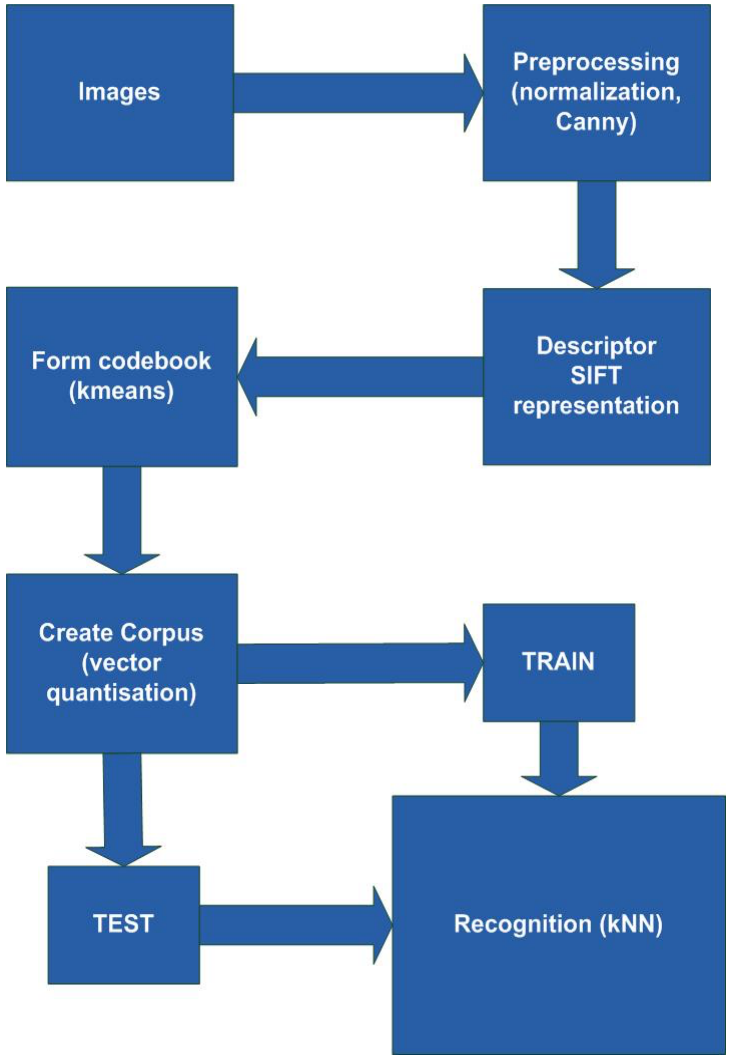
places2.csail.mit.edu, 2017
400+ *scenes*, 10M *images*





Improved Results, 2013 year, 8 categories

All images	2688
Dataset	cvcl.mit.edu/database.htm
Norm size	200x200
Key Points	150 per image
Operator	Canny
Representation	SIFT
Clustering	k-means
Train/Test Algorithm	pLSA
Histogram compare	**kNN method**
Train/Test split	0.5
Categories	8
Words	300
Topics	20
OS	Linux Mint 13 Maya (64b)
Environment	Matlab 2013a (64b)
Overall accuracy	**81,03%**



SIFT

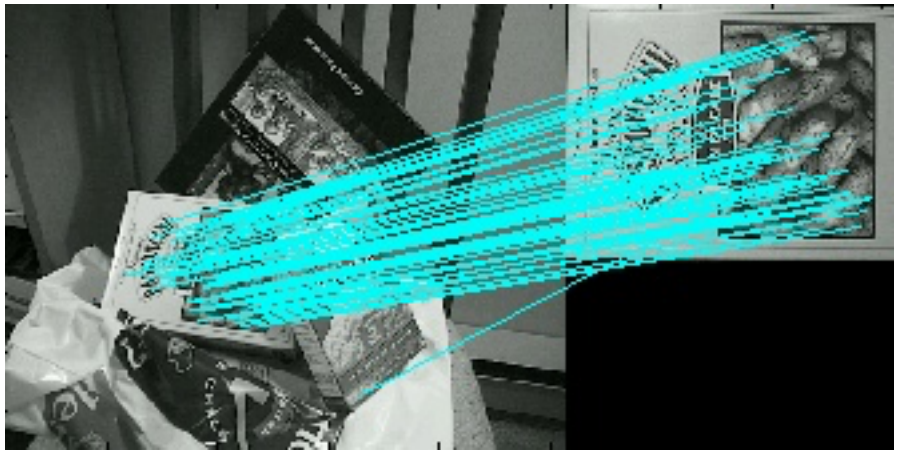
SURF

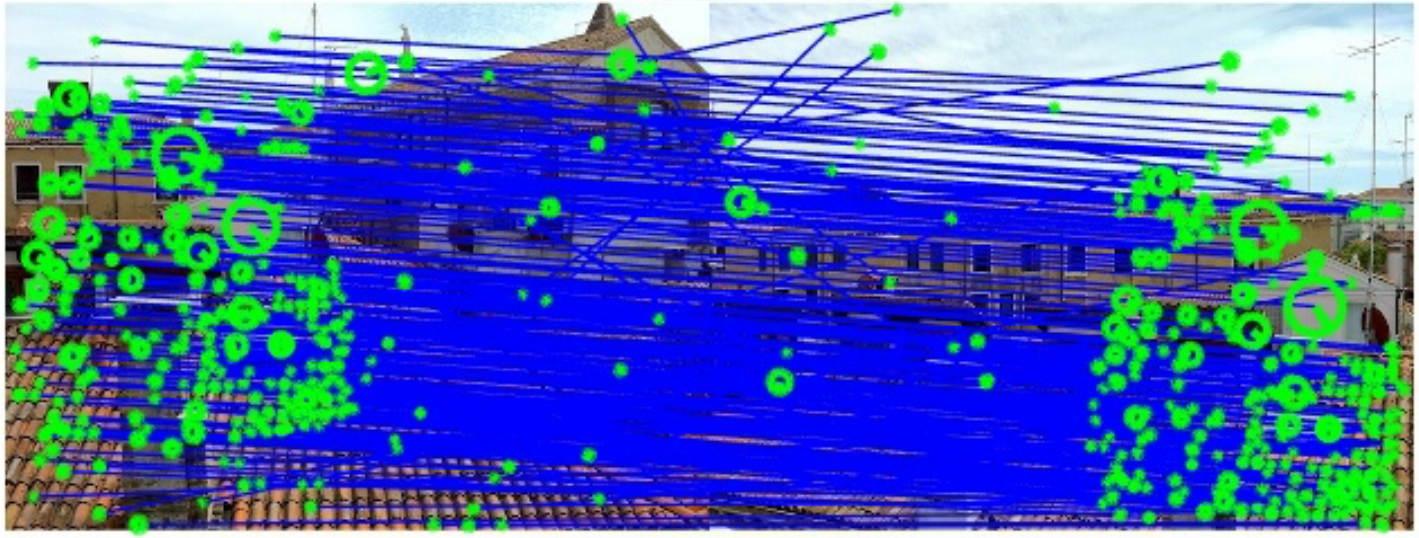
BRIEF

ORB

www.cs.ubc.ca/~lowe/keypoints

www.vlfeat.org/overview/sift.html

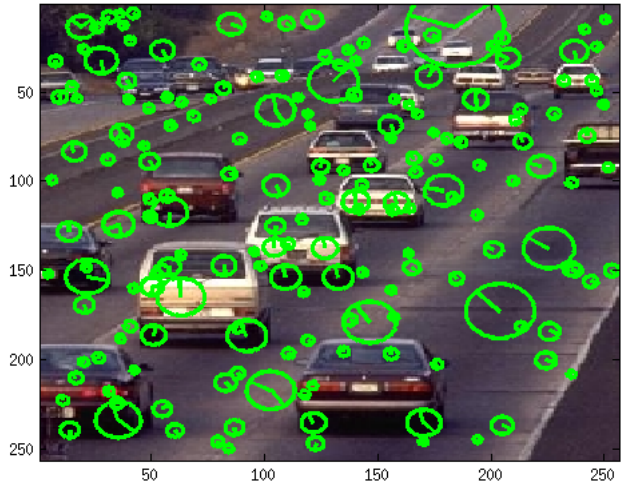


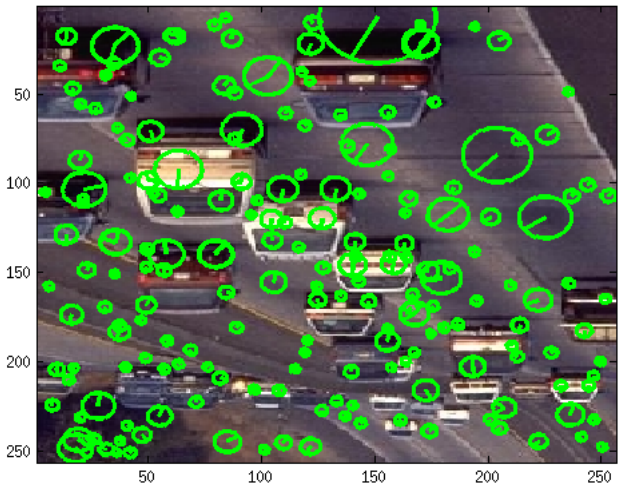


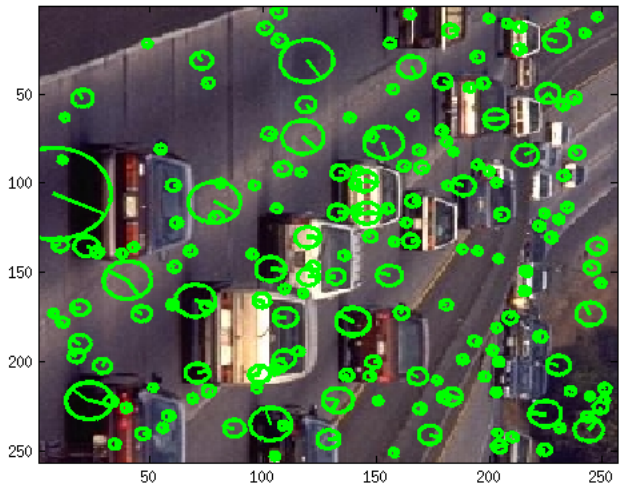
people.ee.ethz.ch/~surf/index.html

docs.opencv.org/master/db/d27/tutorial_py_table_of_contents_feature2d.html

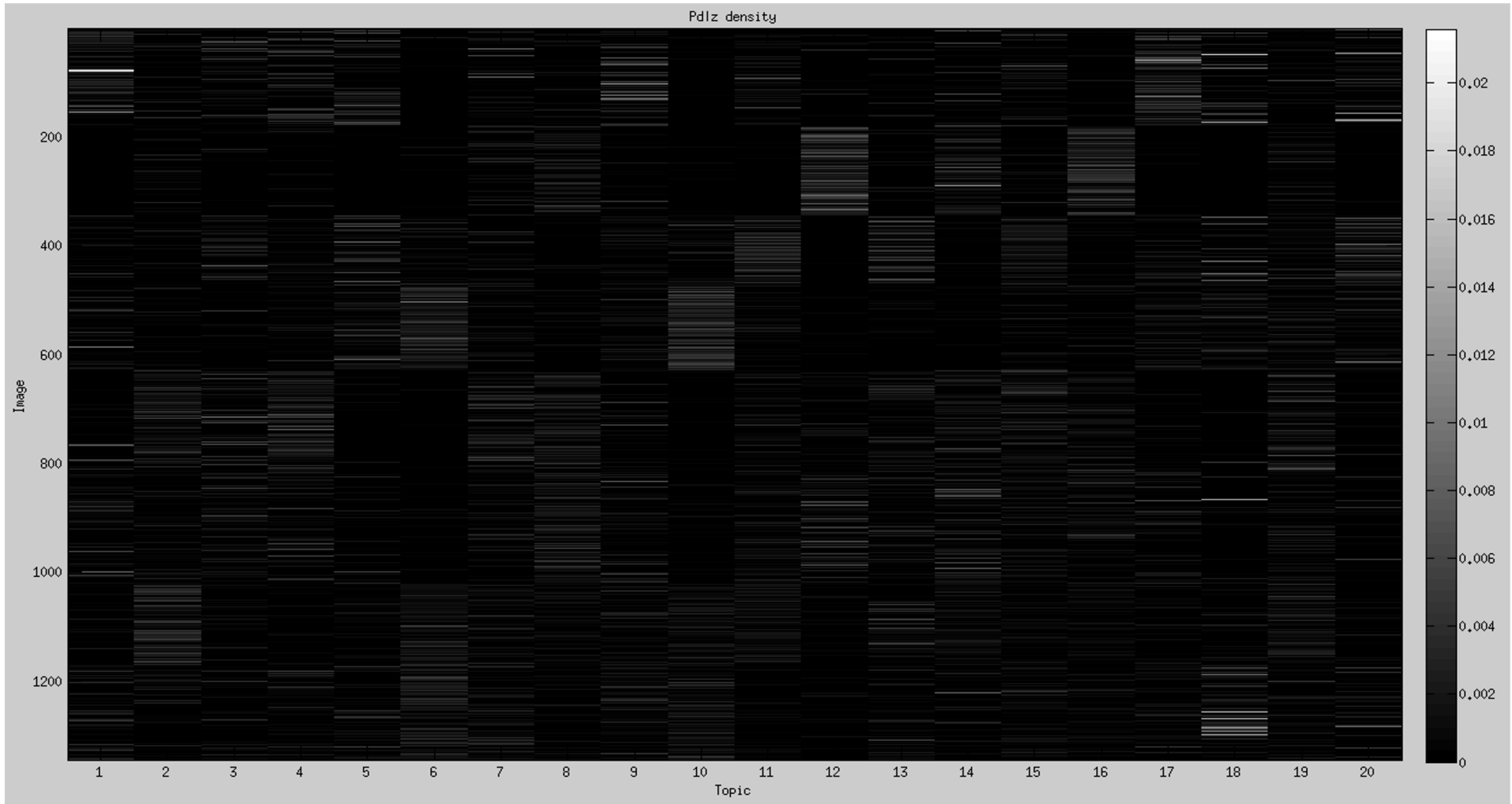
SIFT Rotations



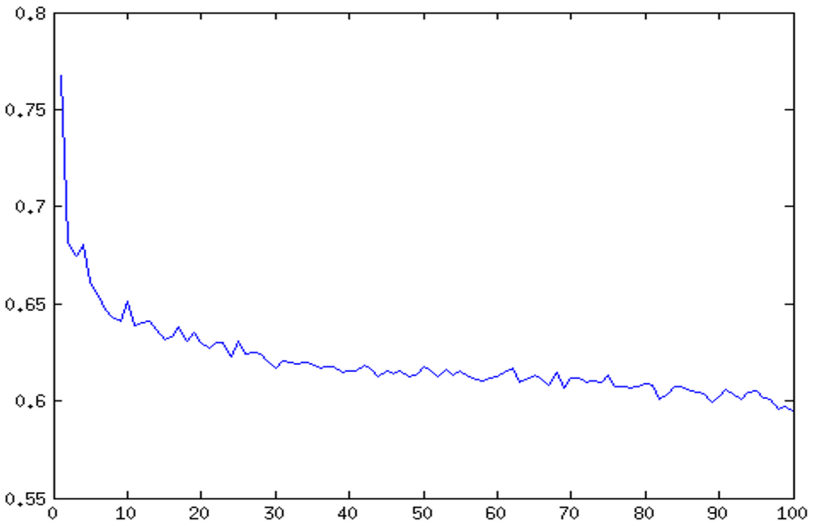




P(d|z) – Image / Topic Intensity



How improve accuracy (kNN)? accuracy as f(k)



euclidean – **standard**

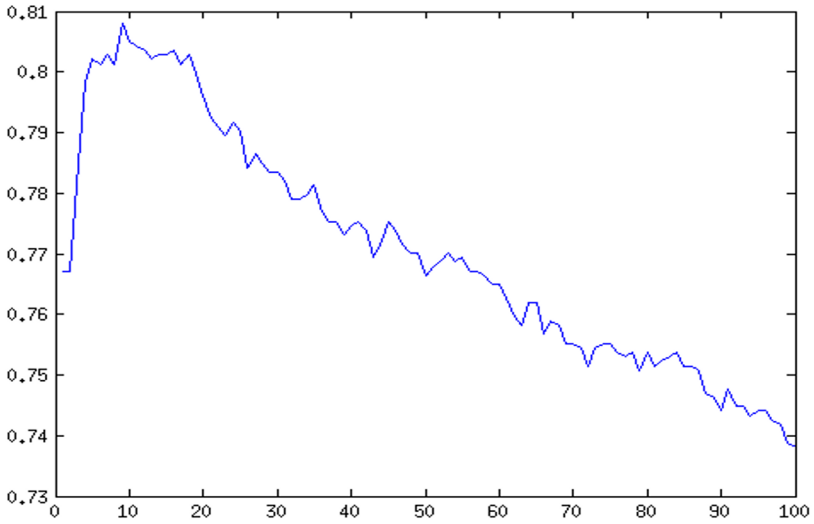
kNN** option – **Distance

euclidean – **standard**
seuclidean
cityblock
chebychev
minkowski
mahalanobis
cosine
correlation
spearman
hamming
jaccard

`DistanceWeight` — Distance weighting function
`'equal'` | `'inverse'` | `'squaredinverse'` | function handle

Distance weighting function, specified as one of the values in this table.

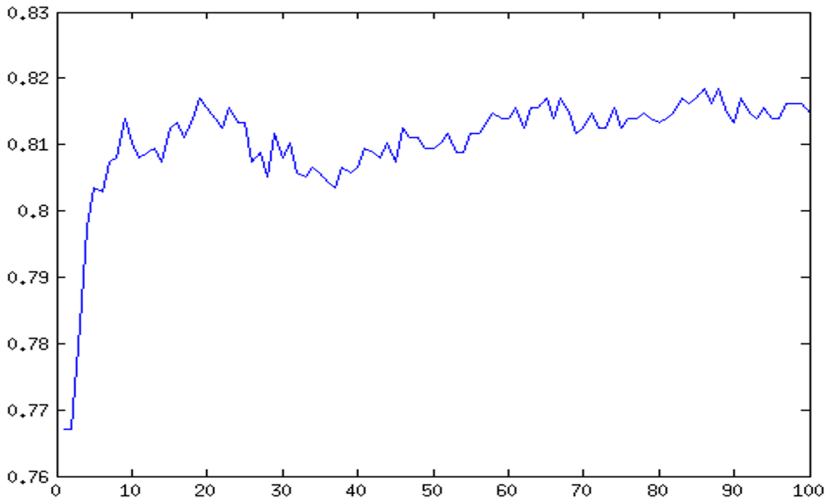
Value	Description
`'equal'`	No weighting
`'inverse'`	Weight is 1/distance
`'squaredinverse'`	Weight is $1/\text{distance}^2$

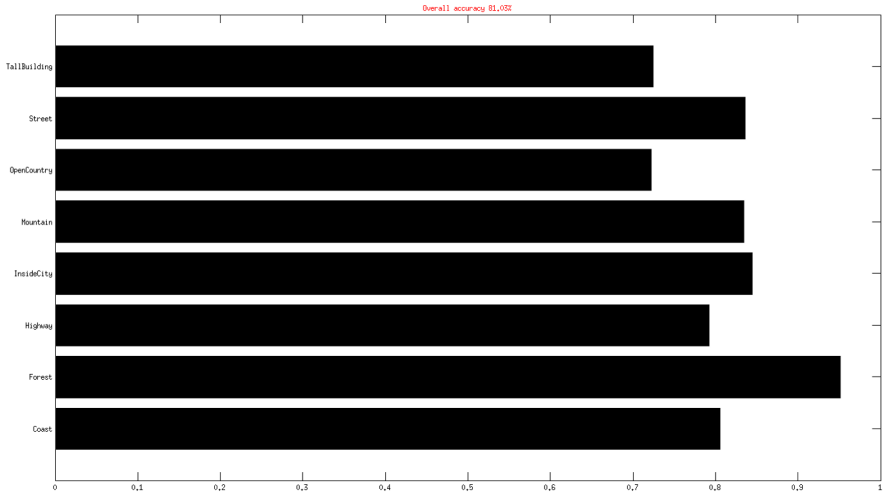


DistanceWeight 'Inverse'

www.mathworks.com/help/stats/classificationknn.html;jsessionid=44bf26fbd9f31fcb294d658a7aad

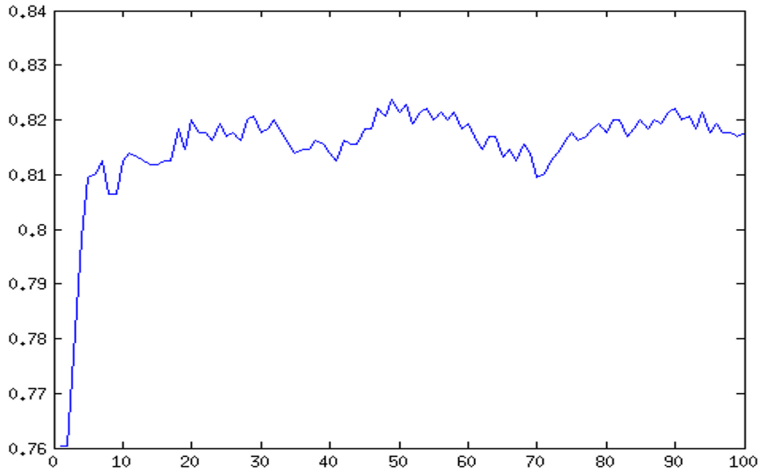
kNN improvements



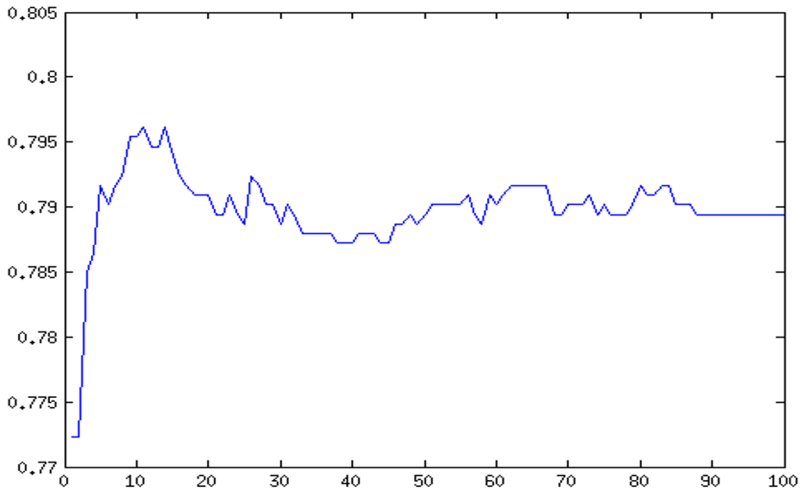


*kNN option – **Distance**, with DistanceWeight set as **SquaredInverse** and **Euclidean Distance***

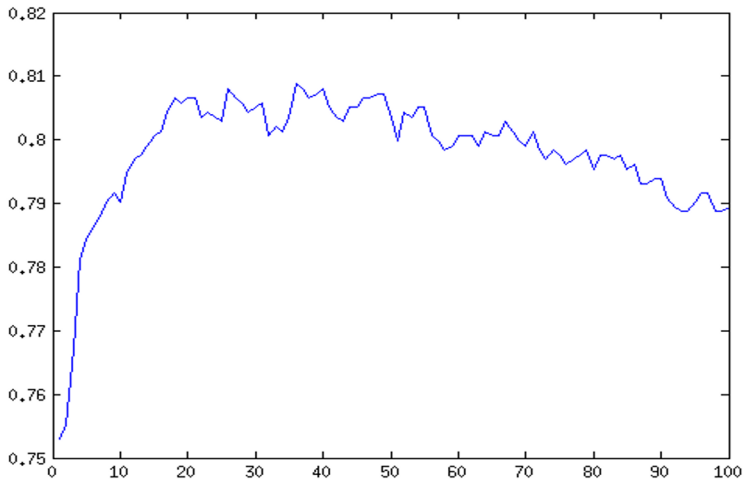
kNN



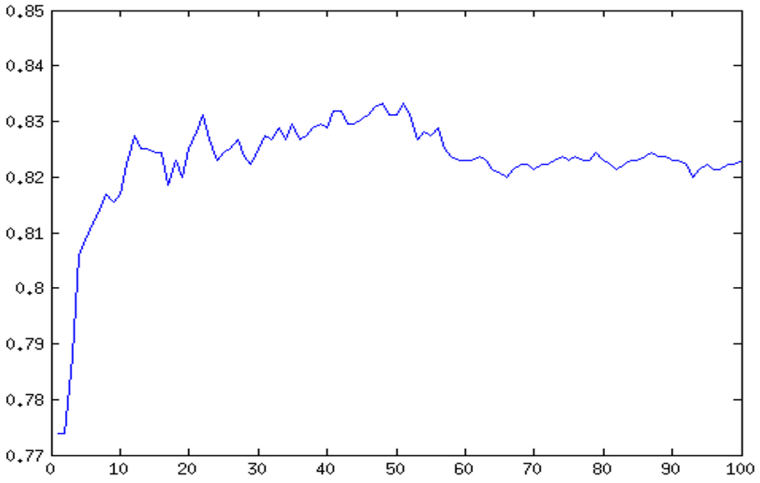
Squared Inverse & Euclidean Distance



Squared Inverse & Correlation

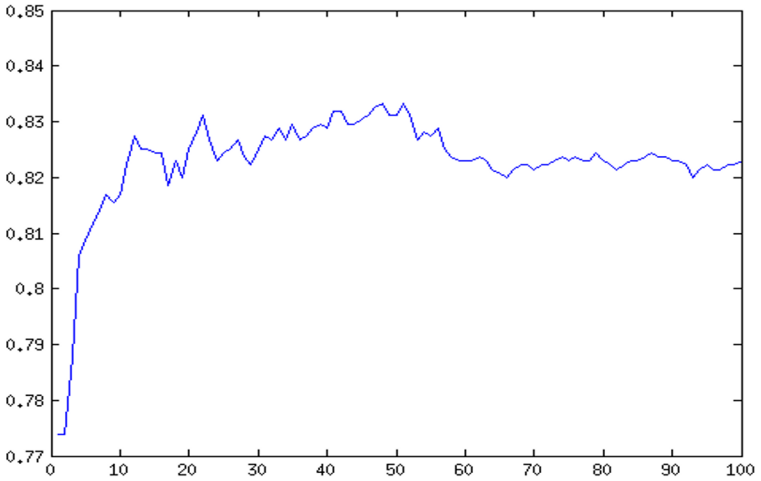


Squared Inverse & Chebychev

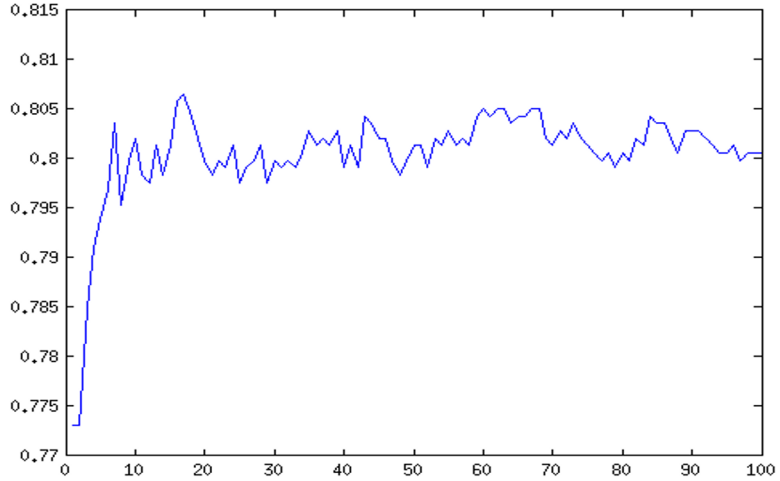


Squared Inverse & Cityblock (BEST)

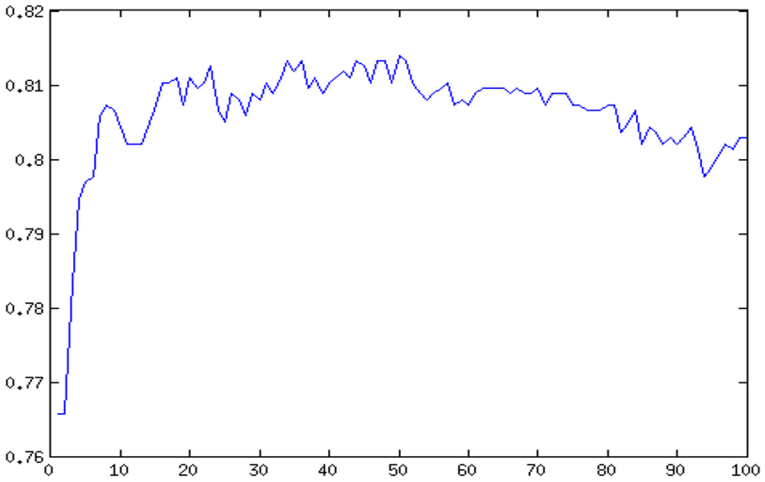
kNN



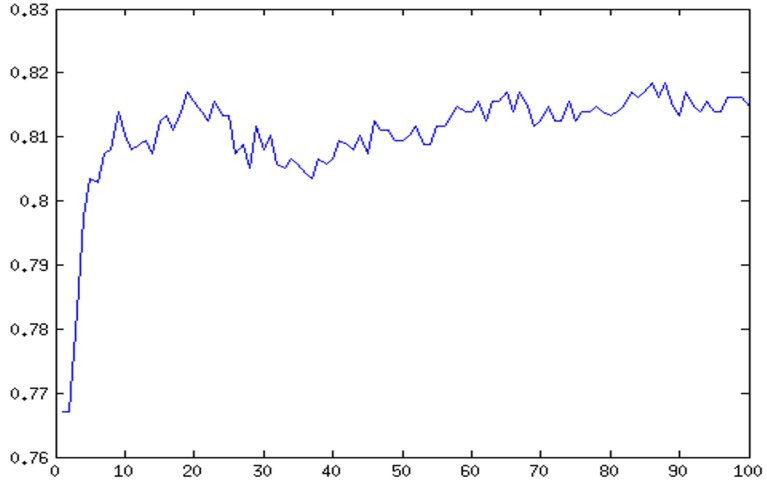
Squared Inverse & Cityblock (BEST)



SquareInverse & Cosine



Squared Inverse & Mahalanobis



SquaredInverse & Minkowski

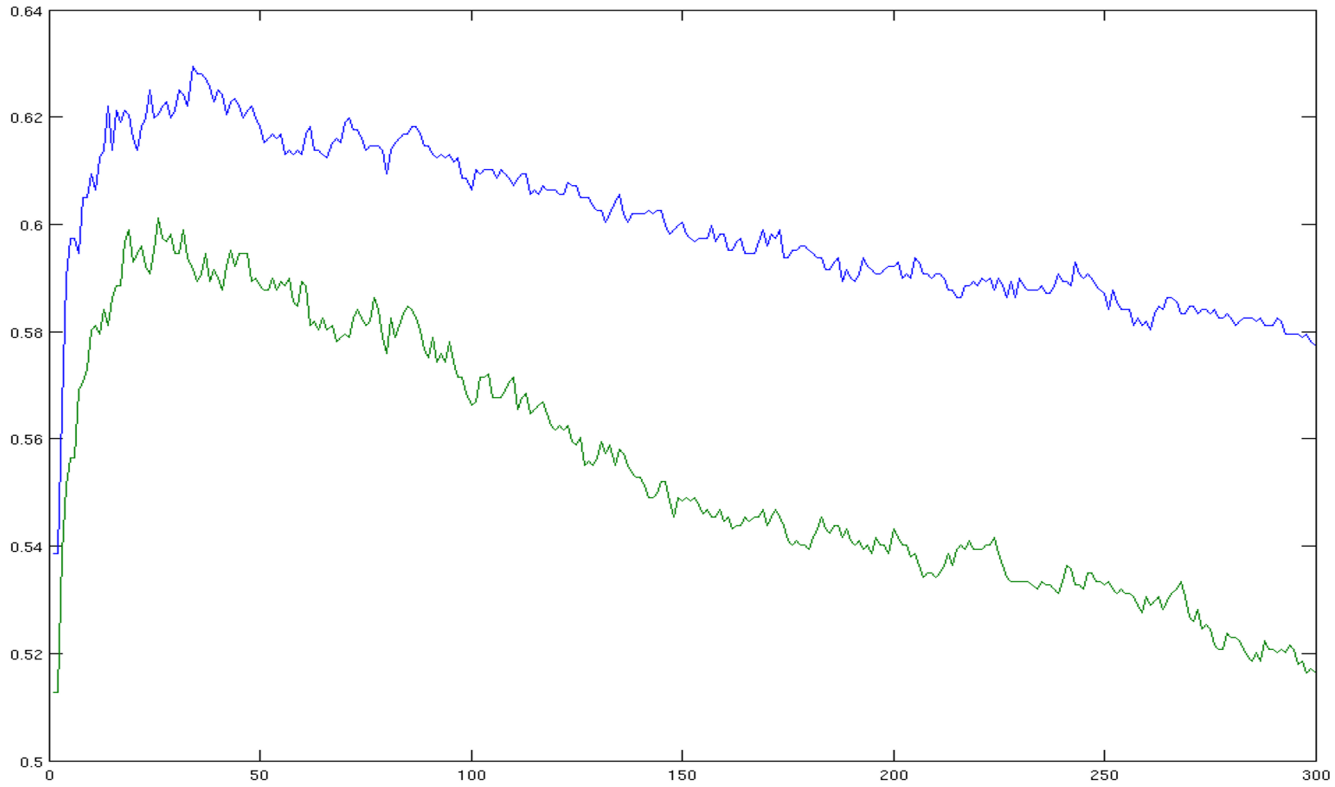
$$D(\mathbf{x}_i, \mathbf{x}_j) = \left(\sum_{l=1}^{d} |x_{il} - x_{jl}|^{1/p} \right)^{p}.$$

MATLAB Profiler

do_all	1	9342.028 s	0.005 s	
do_form_codebook	1	8024.013 s	0.283 s	
do_interest_operator	1	184.820 s	0.000 s	
do_plsa	1	304.365 s	0.260 s	
do_plsa_evaluation	1	20.353 s	0.362 s	
do_preprocessing	1	16.235 s	0.181 s	
do_random_indices	1	0.005 s	0.000 s	
do_representation	1	67.857 s	0.000 s	
do_vq	1	724.369 s	724.109 s	

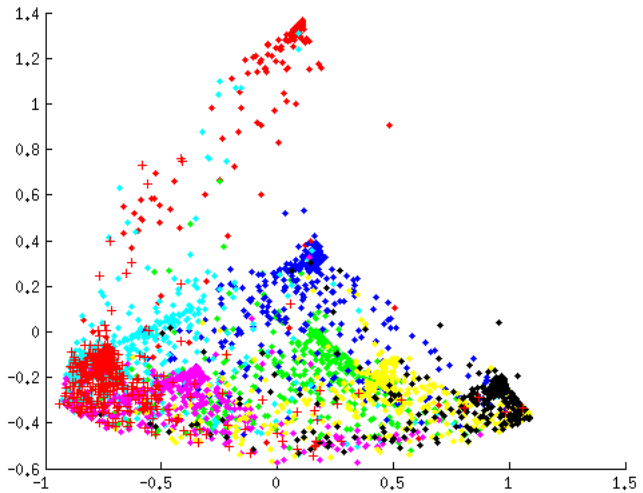
```
  time    calls  line
                 751 function D = distfun(X, C, dist, iter,rep, reps)
                 752 %DISTFUN Calculate point to cluster centroid distances.
< 0.01     1540   753 [n,p] = size(X);
 59.64     1540   754 D = zeros(n,size(C,1));
  0.02     1540   755 nclusts = size(C,1);
                 756
  0.02     1540   757 switch dist
< 0.01     1540   758     case 'sqeuclidean'
           1540   759         for i = 1:nclusts
 86.89   320198   760             D(:,i) = (X(:,1) - C(i,1)).^2;
  1.02   320198   761             for j = 2:p
7581.16 40665146   762                 D(:,i) = D(:,i) + (X(:,j) - C(i,j)).^2;
 84.36 40665146   763             end
                 764             % D(:,i) = sum((X - C(repmat(i,n,1),:)).^2, 2);
  0.67   320198   765         end
                 766     case 'cityblock'
```

kNN (Euclidean vs CityBlock)

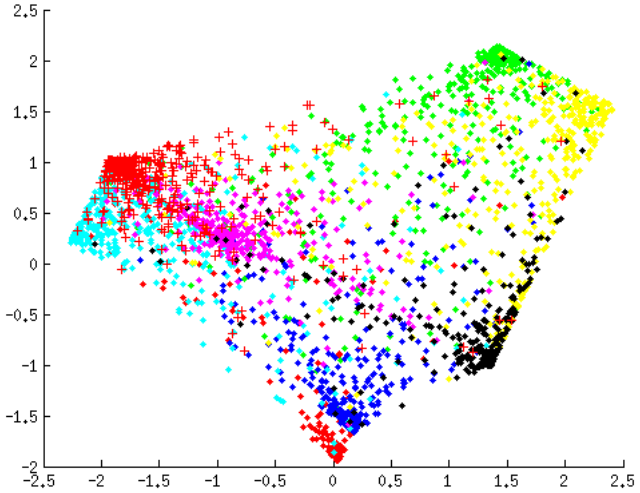


accuracy with kNN, k = 1..300. **Green – Euclidean**, **Blue – Cityblock**

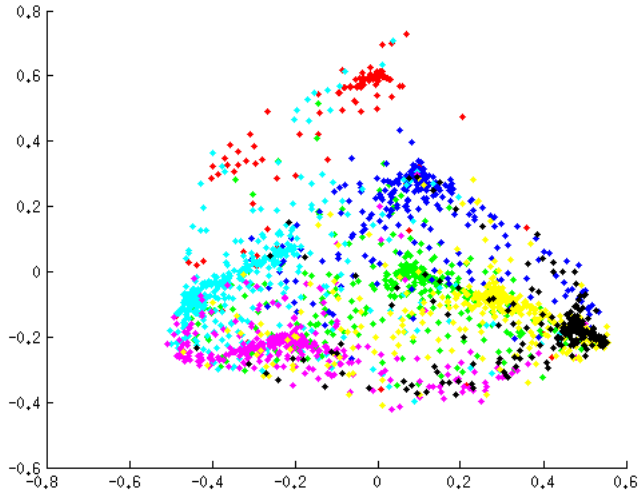
Class Visualization



TrueCityBlock

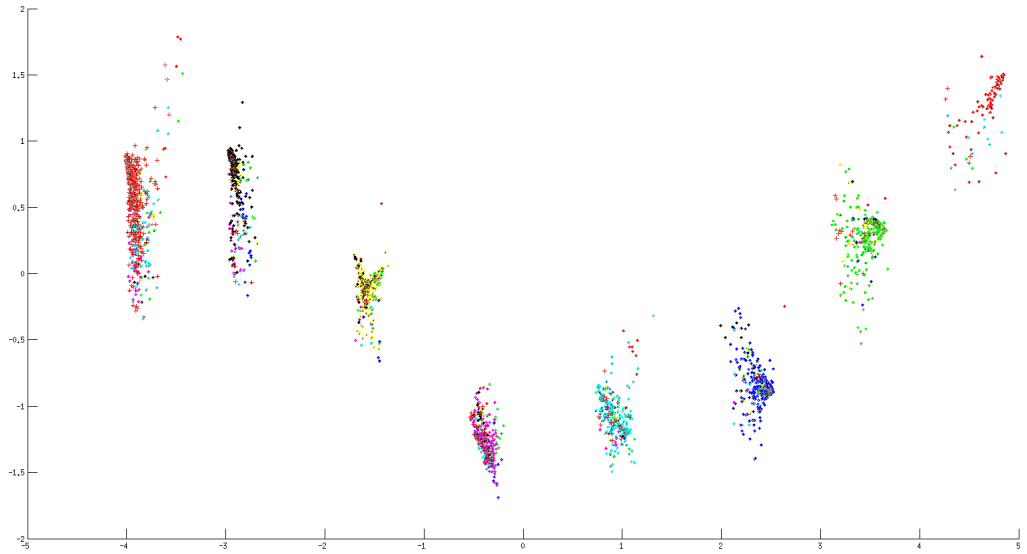


TrueSEuclidean



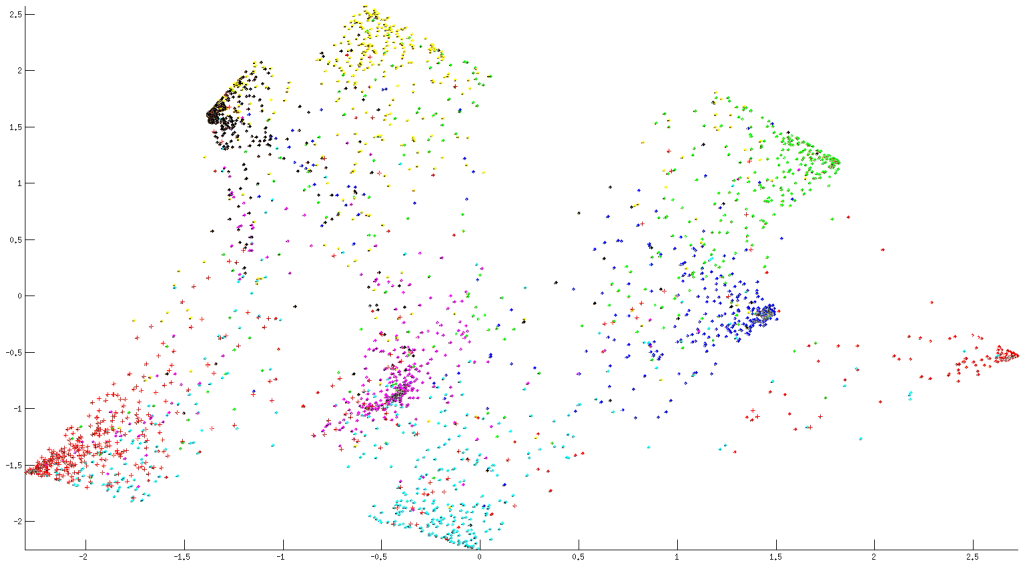
TrueCosine

Class Visualization

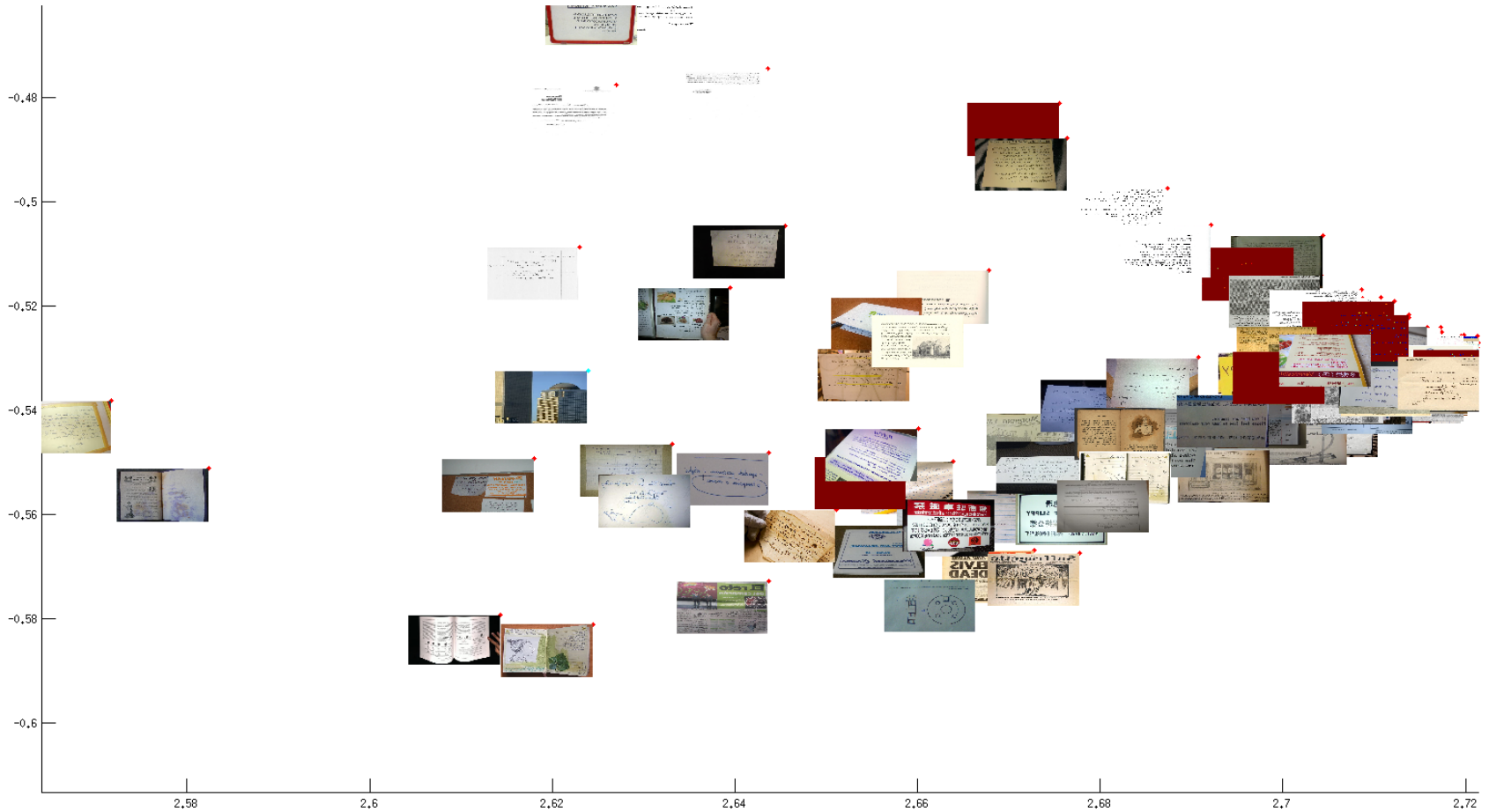


ClassesCityBlock

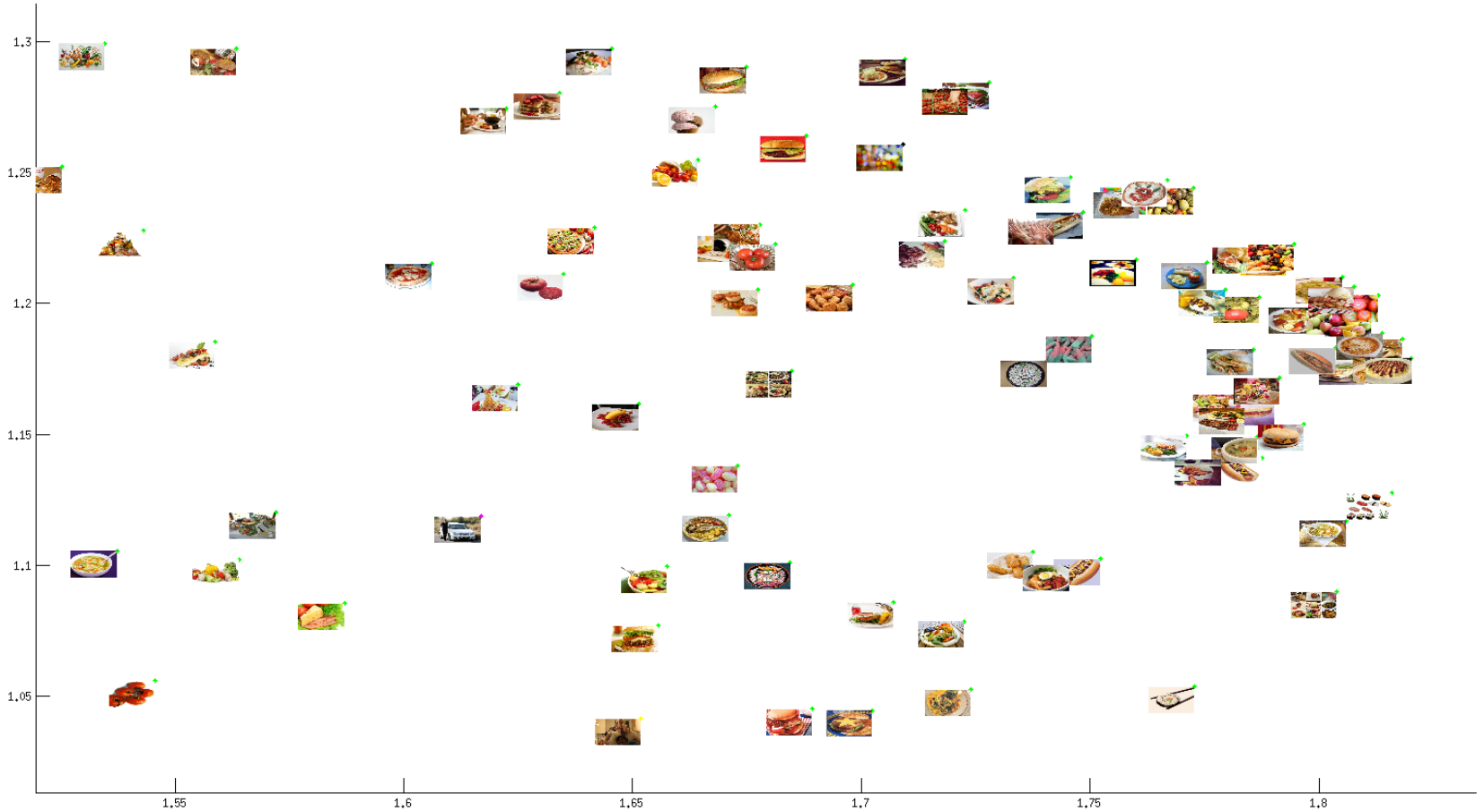
ClassesStandardEuclidean



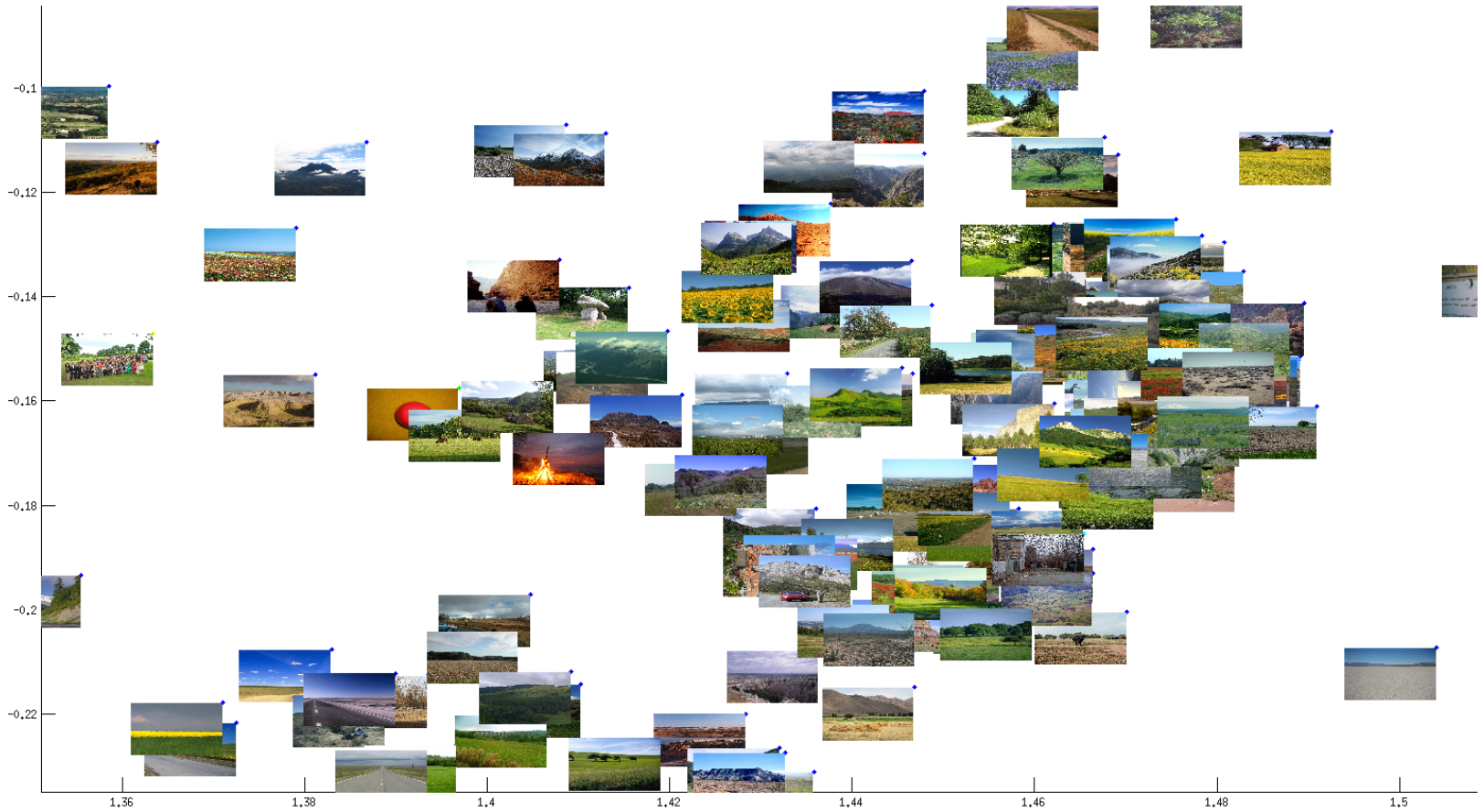
Groups - Text



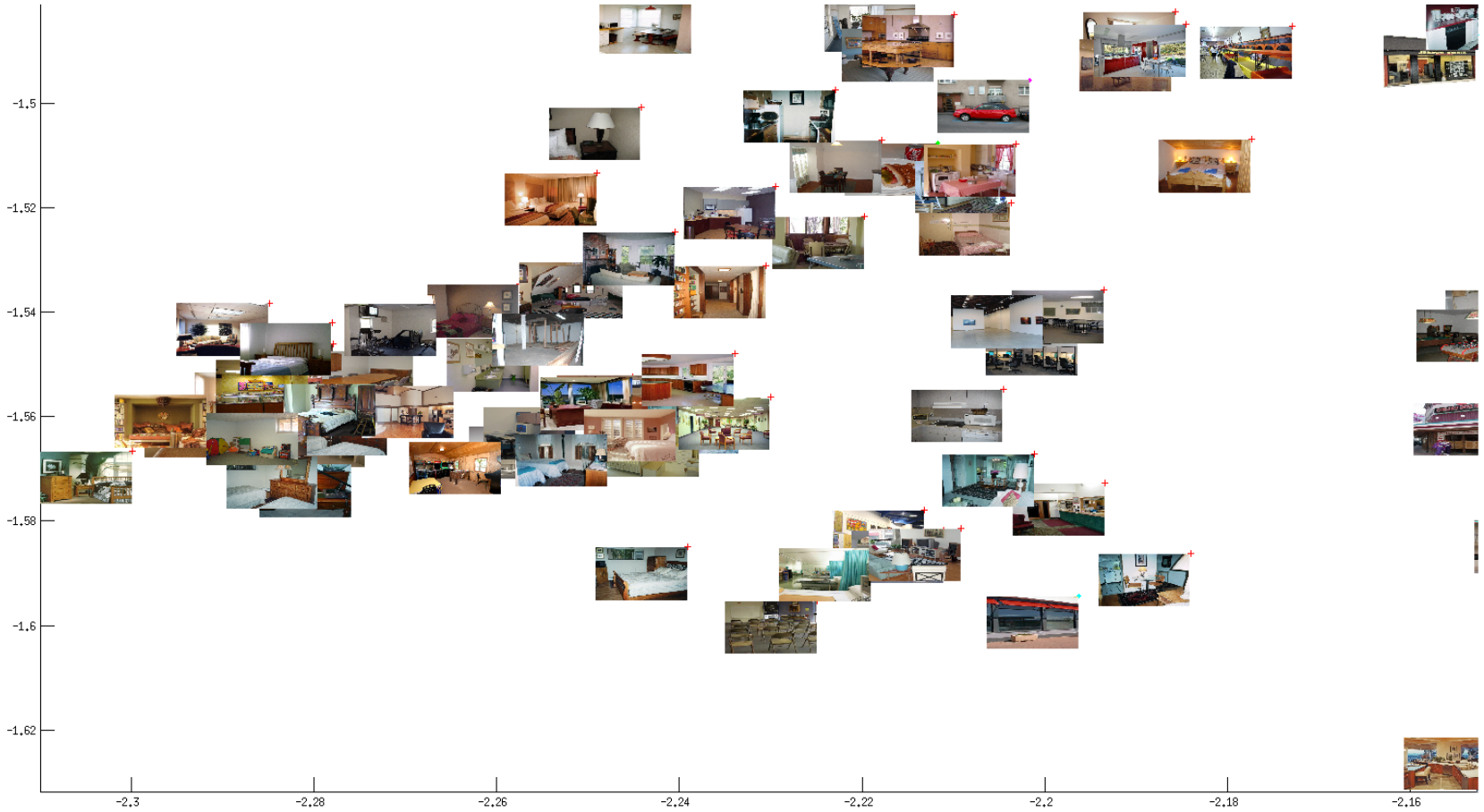
Groups - Food



Groups - Landscape



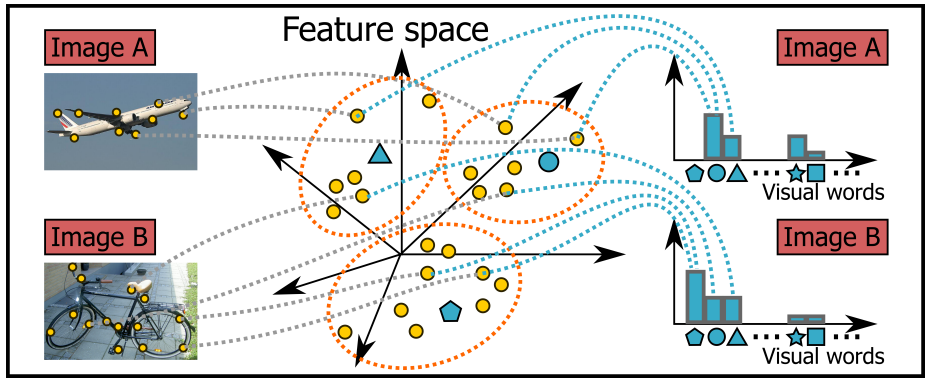
Groups - Indoors

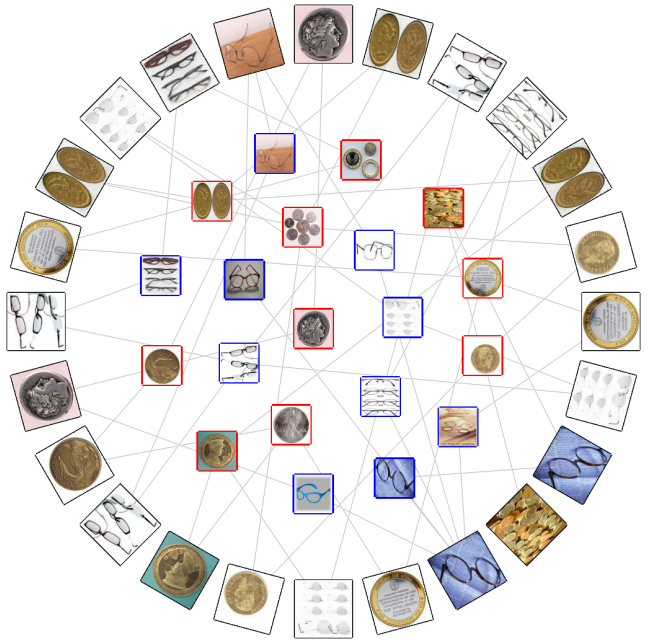


Groups - Night

Groups - Cars

Visualizing Bag-of-Features Image Categorization Using Anchored Maps





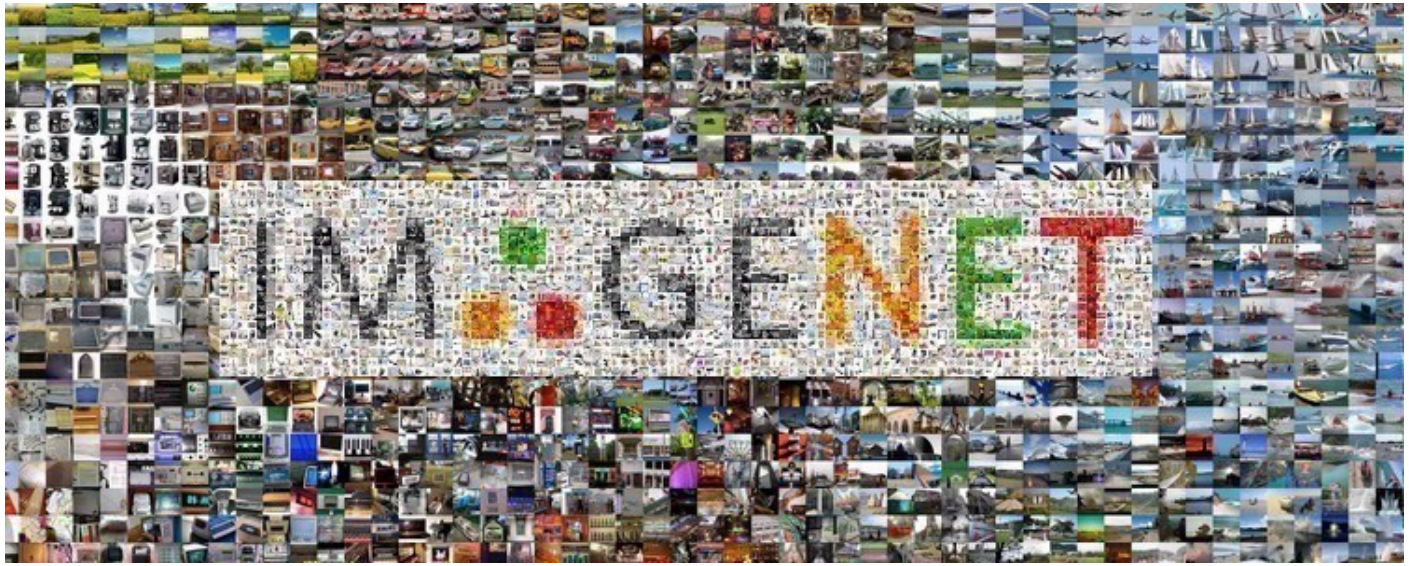
www.u-aizu.ac.jp/~shigeo/research/explore/index-e.html
youtube.com/watch?v=jtxFW3CYclI

Caffee Toolbox

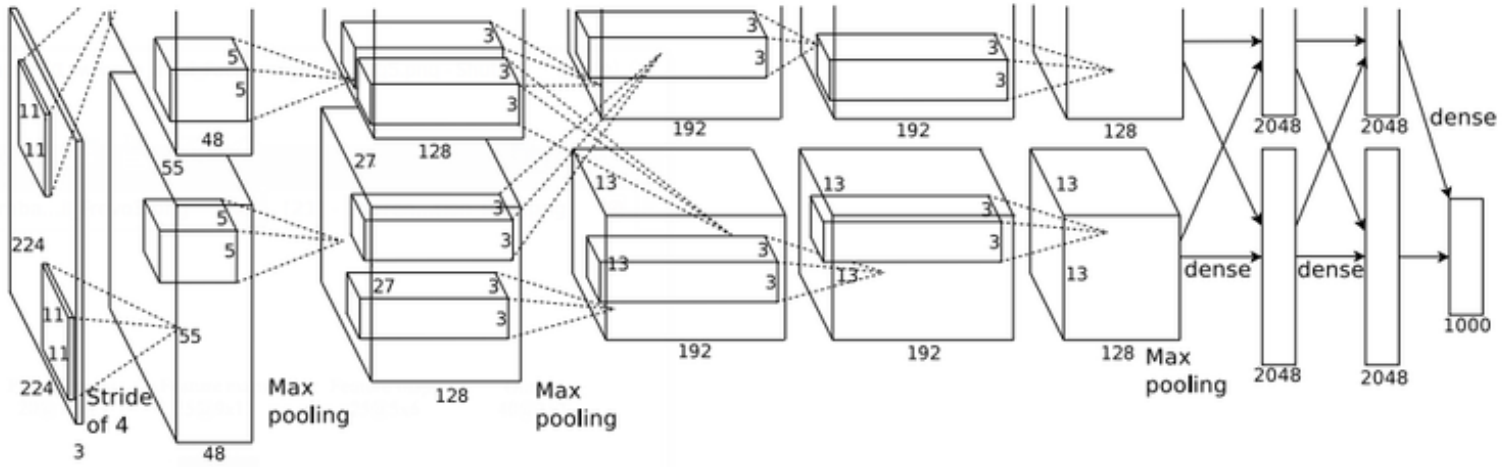
github.com/BVLC/caffe

daggerfs.com

arxiv.org/abs/1310.1531



AlexNet, 2012



https://www.cs.toronto.edu/~kriz