CS231n

Lecture 1,2 summary

History of computer vision study

Effort to recognize object

- Simplify
- Block World(Larry Roberts,1963)
- Stages of Visual Representation(David Marr,1970s)
- Generalized Cylinder(Brooks&Binford, 1979)
- Pictorial Structure(Fischler and Elschlager, 1973)
- Constructing Line and Edges(David Lowe, 1987)

→ object recognition was difficult. New approach image seg.

- Normalize cut (Shi & Malik, 1997)

History of computer vision study

Effort to recognize object

- Feature based recognition was tried.
- Machine Learning Algorithm(SVM, Boosting, CNN,Etc) appeared

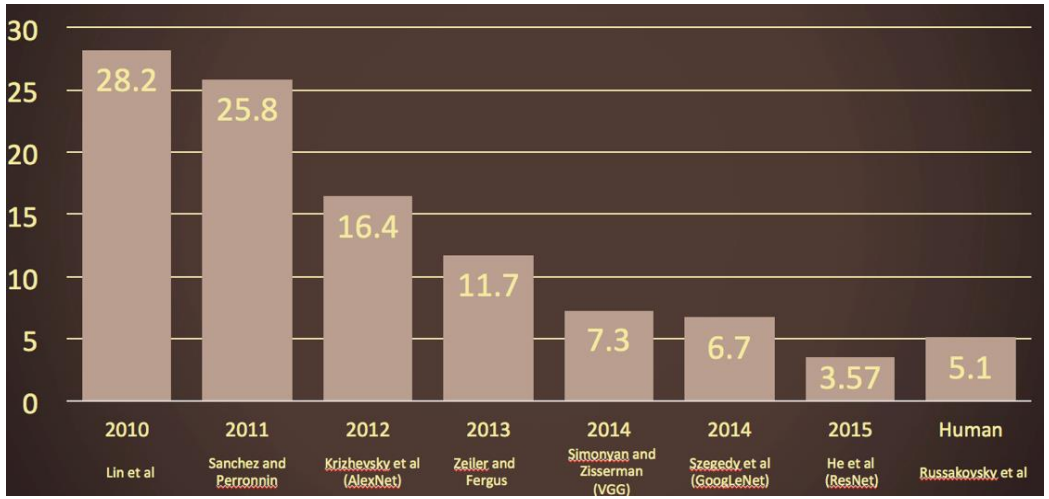
→ face detection..

- “SIFT” & Object Recognition (David Lowe, 1999)
- Spatial Pyramid Matching(Lazebnik, Schmid & Ponce, 2006)
- Histogram of Gradients (HoG) (Dalal & Triggs, 2005)
- Deformable Part Model (Felzenswalb, McAllester, Ramanan, 2009)

Some changes

- Quality of Pictures
- Computing power
- Large Data

→ CNN have become an important tool for object recognition



(Lec 2) Image Classification

- Core task in computer vision
- Find fit label in label sets.
- Easy to human, hard to machine
- Semantic Gap

Challenges

- Viewpoint variation
- Illumination
- Deformation
- Occlusion
- Background Clutter
- Intraclass variation

→NO obvious algorithm

Data Driven Approach

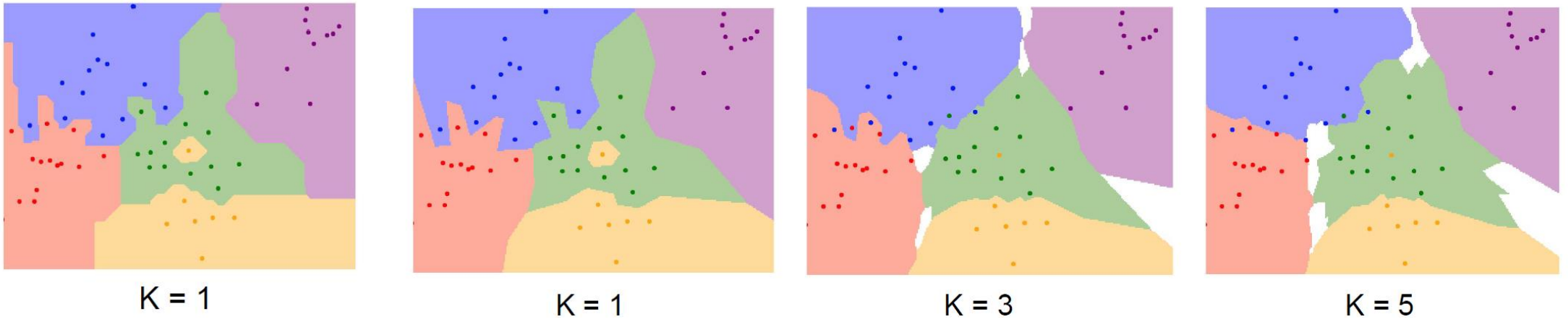
- 1. Collect a dataset of images and labels
- 2. Use Machine Learning to train a classifier
- 3. Evaluate the classifier on new images

```
def train(images, labels):
  # Machine learning!
  return model
```

```
def predict(model, test_images):
  # Use model to predict labels
  return test_labels
```

First Classifier : Nearest Neighbor

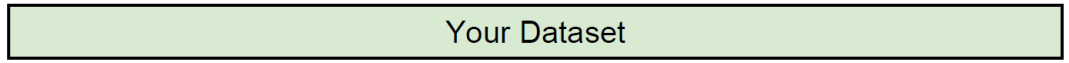
- Metric distance : L1,L2
- Train : O(1) Predict O(N)
- K-Nearest Neighbor : Take majority vote from K closest points
- Hyperparameter : K value, Metric define.



Setting hypermeter

Idea #1: Choose hyperparameters that work best on the data

BAD: K = 1 always works perfectly on training data



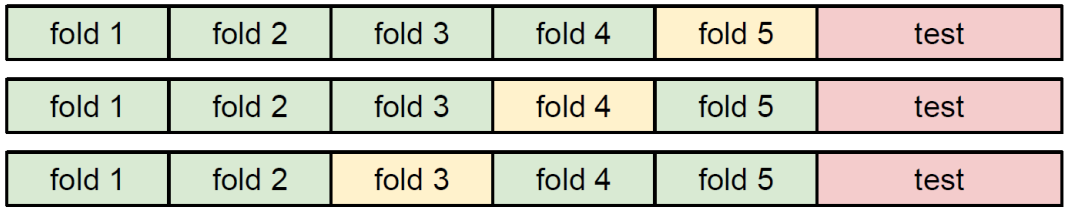
Idea #2: Split data into **train** and **test**, choose hyperparameters that work best on test data

BAD: No idea how algorithm will perform on new data

Idea #3: Split data into **train**, **val**, and **test**; choose hyperparameters on val and evaluate on test

Better!

Idea #4: **Cross-Validation**: Split data into **folds**, try each fold as validation and average the results



Useful for small datasets, but not used too frequently in deep learning

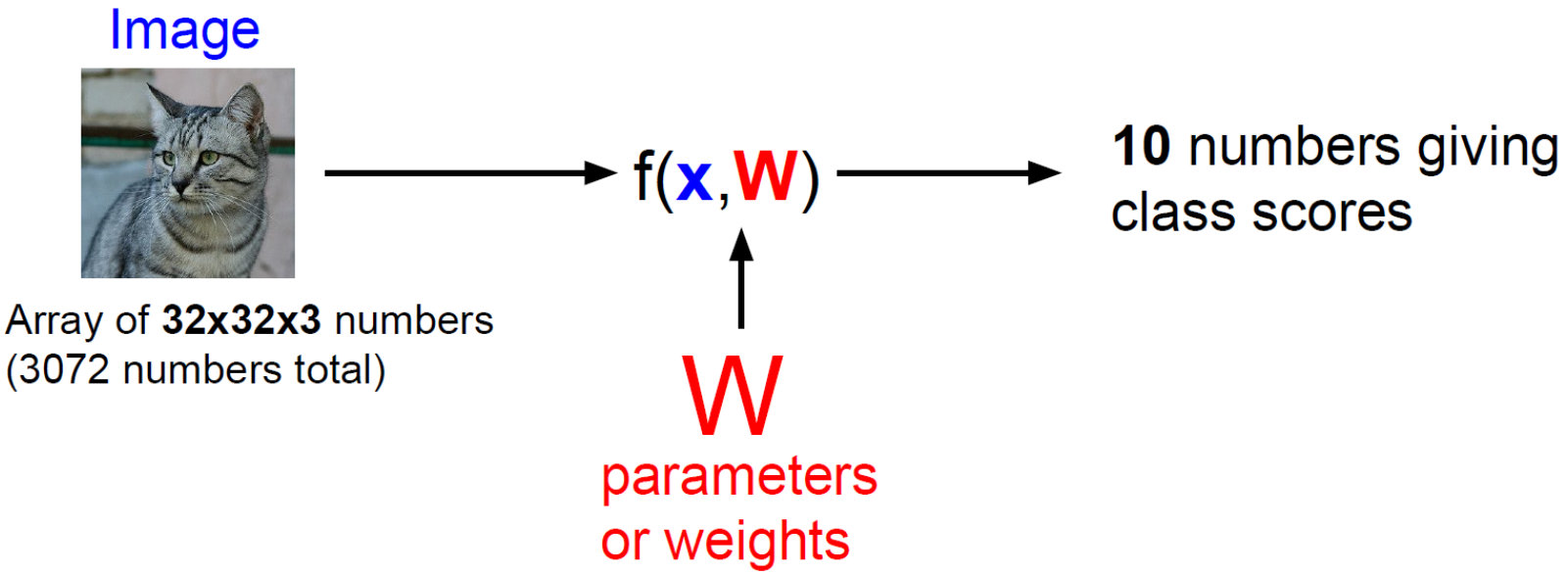
k-Nearest Neighbor on images **never used.**

- Very slow at test time
- Distance metrics on pixels are not informative
- Curse of dimensionality

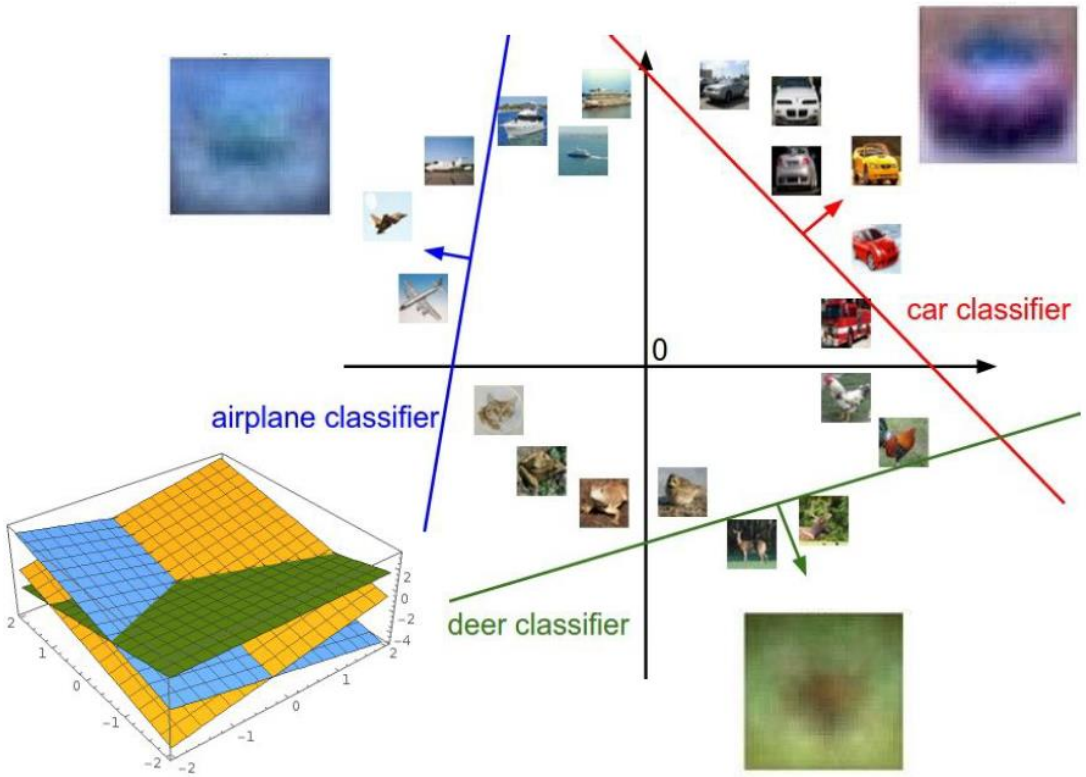
Linear Classification

- Components of neural networks

$$f(x, W) = Wx + b(ias) \quad f: 10by1, W: 10by3072, x: 3072by1\ , b: 10by1$$



Linear Classification



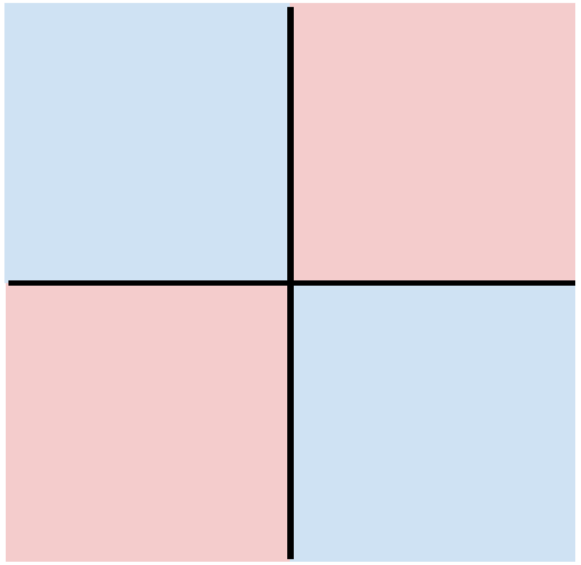
$$f(x,W) = Wx + b$$

Array of **32x32x3** numbers
(3072 numbers total)

Hard Case for a linear classification

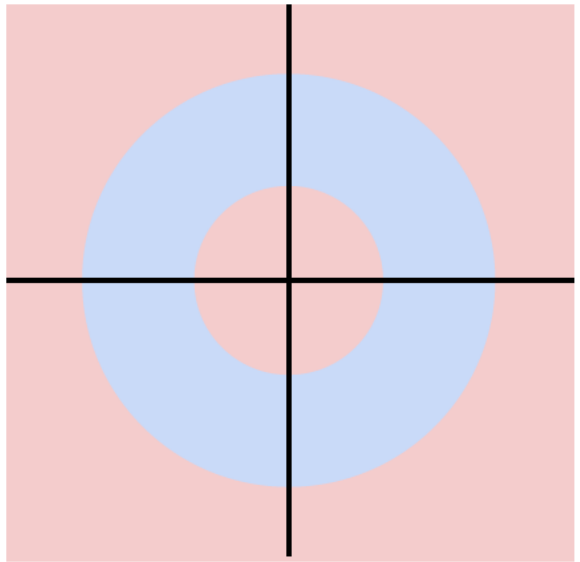
Class 1:
number of pixels > 0 odd

Class 2:
number of pixels > 0 even



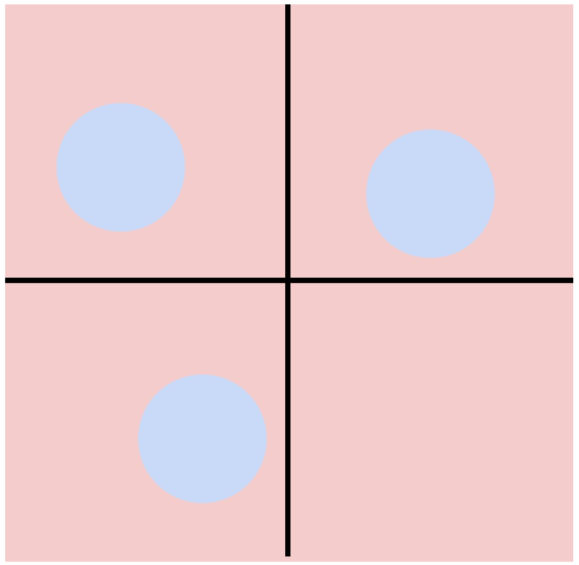
Class 1:
1 <= L2 norm <= 2

Class 2:
Everything else



Class 1:
Three modes

Class 2:
Everything else



Define f(x,W)

- Different performance to different class -> Is this good or bad?



airplane	-3.45	-0.51	3.42
automobile	-8.87	**6.04**	4.64
bird	0.09	5.31	2.65
cat	**2.9**	-4.22	5.1
deer	4.48	-4.19	2.64
dog	8.02	3.58	5.55
frog	3.78	4.49	**-4.34**
horse	1.06	-4.37	-1.5
ship	-0.36	-2.09	-4.79
truck	-0.72	-2.93	6.14

Thank you