

②

Image Classification pipelineImage Classification

↳ A Core task in Computer Vision

⇒ Assume given set of discrete labels

Semantic Gap

⇒ 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 image.

① Nearest Neighbor Classifier

def train(images, labels):

⑥ # Machine learning,
return model

→ Memorize all
data & labels

def predict(model, test_images):

⑥ # Use model to predict labels
return test_labels

→ Predict the label
of the most
similar training
image

⇒ Distance Metric to Compare Images

① L1 distance: $d_1(I_1, I_2) = \sum_p |I_1^p - I_2^p|$

{also called Manhattan distance}

② K-Nearest Neighbors

⇒ Instead of copying label from nearest neighbor, take majority vote from K closest points.

⇒ L2 (Euclidean) distance

$$d_2(I_1, I_2) = \sqrt{\sum_p (I_1^p - I_2^p)^2}$$

Hyperparameters: Choices about the algorithm that we set rather than learn.

- Example: K , distance function.

⇒ K-Nearest Neighbor on images never used

↳ Very slow at test time

↳ Distance metrics on pixels are not informative.

* Linear Classification

Parametric Approach

Image
 $32 \times 32 \times 3$

→ $f(x, W)$

10 numbers
giving class
scores

Parameters
on
weights

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

$$f(x, W) = \overset{b+}{W} x \rightarrow 3072 \times 1$$

\downarrow
 10×3072