Lecture Notes on CS231N

Yiwei Chen

Zhejiang University

April 4, 2025

Lecture 1: History of CV and Introduction to CNNs

- **ImageNet:** Annual competition for image classification, started in 2010.
- **Convolutional Neural Networks (CNNs):** Introduced by Yann LeCun in 1998, CNNs are a type of neural network designed for processing structured grid data, such as images. CNNs show great performance in image classification tasks.

Lecture 2: Image Classification Pipeline

1. **Attempts:**
   - Find edges, then corners: does not work well.
   - Use large datasets with labels.

2. **Classifiers:**

   (a) **K-Nearest Neighbors (KNN):**

   - *Description:* When $K = 1$, Find the closest image in the dataset to the input image (Nearest Neighbors (NN)).
   - *Distance metric:*

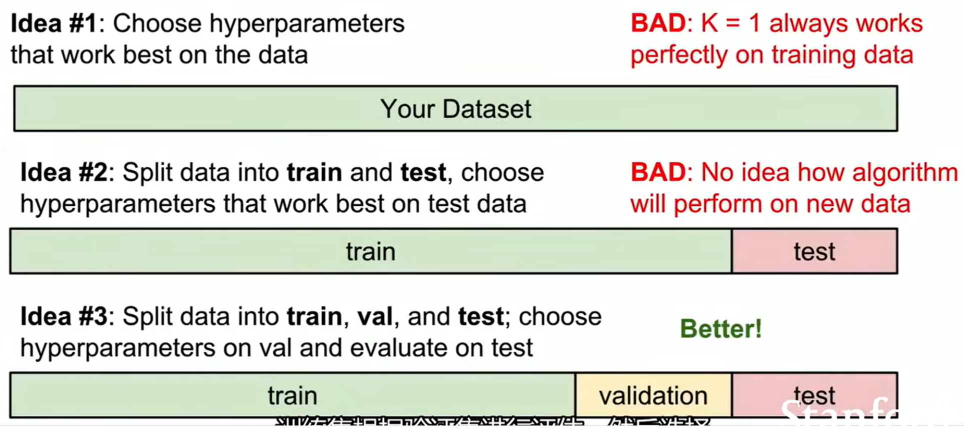
     i. L1(Mahanttan) Distance: a squared distance metric.

     $$d(x, y) = \sum_i |x_i - y_i| \tag{1}$$

     ii. L2(Euclidean) Distance: a squared distance metric.

     $$d(x, y) = \sqrt{\sum_i (x_i - y_i)^2} \tag{2}$$

     Rotating the coordinate system changes the L1 distance but not the L2 distance.
   - *Performance:*
     Training time: $O(1)$, as there is nothing to do.
     Prediction time: $O(N)$, which is inefficient.
   - K-Nearest Neighbors (KNN):
     Description:
     When $K = 1$, the classifier is too sensitive to noise.
     Instead of copying the label of the closest image, take the majority vote of the K closest images.
   - *hyperparameters*(超参数):
     Choices about the model that are not learned from the data, e.g., K in KNN.
     To set hyperparameters:
     - Never use the test set to set hyperparameters.
     - Splitting data into train and test is not enough.
     - **The better idea:** Splitting the training set into training set, validation set, and test set.



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

BAD: K = 1 always works perfectly on training data

Your Dataset

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

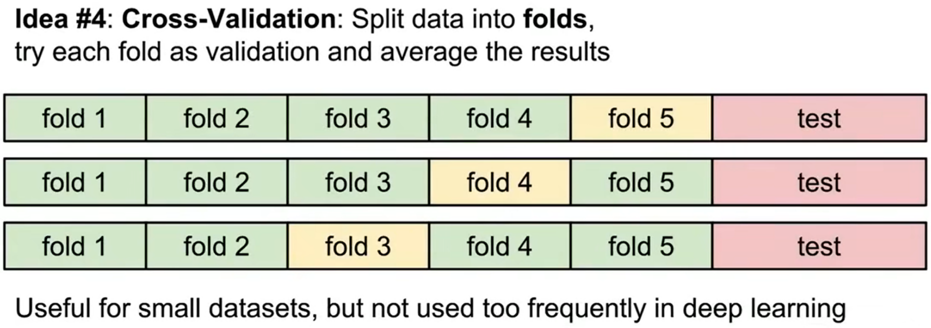
train	test

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

Better!

train	validation	test

– **The common idea:** Cross-validation(交叉验证).



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

fold 1	fold 2	fold 3	fold 4	fold 5	test
fold 1	fold 2	fold 3	fold 4	fold 5	test
fold 1	fold 2	fold 3	fold 4	fold 5	test

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

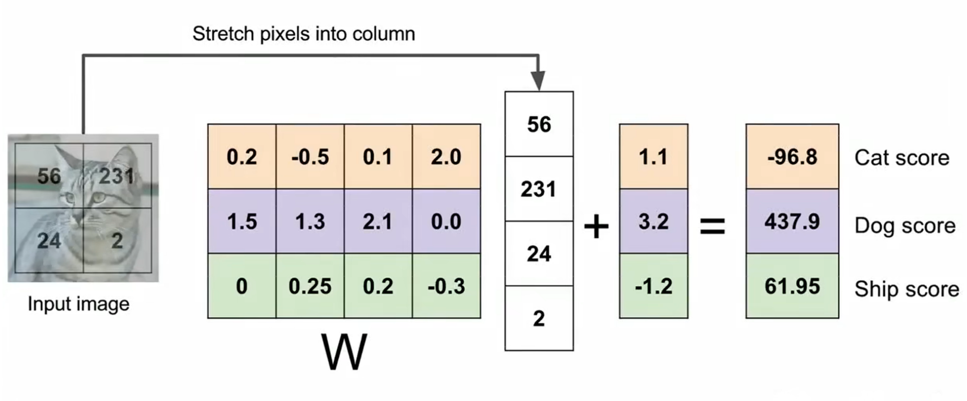
- *Pros and Cons:*
  Actually, KNN on image is never used:
  - Very slow at test time.
  - Distance-metrics on pixels are not informative.
  - Curse of dimensionality: as the number of dimensions increases, the distance between points becomes less meaningful.

(b) **Linear Classifier:**

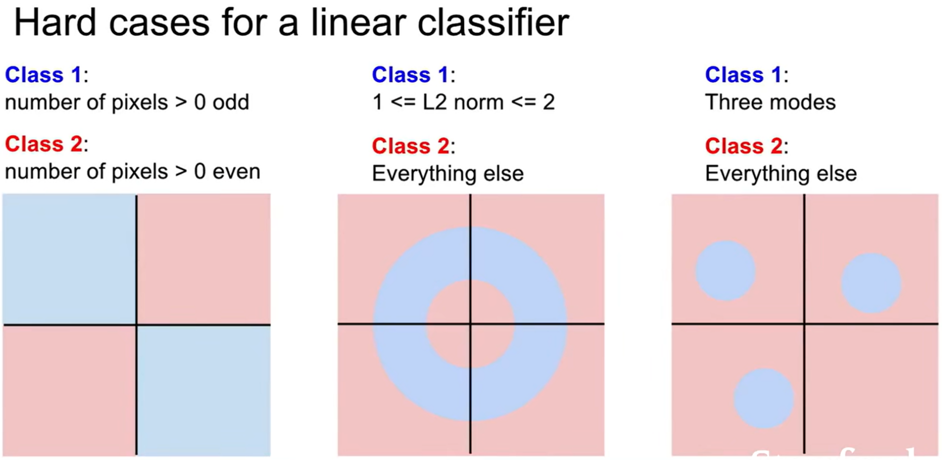
- *Description:* A linear classifier makes its predictions based on a linear predictor function combining a set of weights with the feature vector.

$$f(x, W) = Wx + b \tag{3}$$

where x is the input image and w is the weight vector, b is the bias term.



- *Hard cases:*



Hard cases for a linear classifier

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