CS4670/5670: Computer Vision
Kavita Bala

Lecture 27: Recognition Basics

Slides from Andrej Karpathy and Fei-Fei Li
http://vision.stanford.edu/teaching/cs231n/
Announcements

• PA 3 Artifact voting
  Vote by Tuesday night
Today

• Image classification pipeline
• Training, validation, testing
• Score function and loss function

• Building up to CNNs for learning
  – 5-6 lectures on deep learning
Image Classification

(assume given set of discrete labels)
{dog, cat, truck, plane, ...}

→ cat
Image Classification: Problem

What the computer sees

82% cat
15% dog
2% hat
1% mug

image classification
Data-driven approach

• Collect a database of images with labels
• Use ML to train an image classifier
• Evaluate the classifier on test images

Example training set

Slides from Andrej Karpathy and Fei-Fei Li
http://vision.stanford.edu/teaching/cs231n/
Data-driven approach

• Collect a database of images with labels
• Use ML to train an image classifier
• Evaluate the classifier on test images
Train and Test

• Split dataset between training images and test images

• Be careful about inflation of results
Classifiers

- Nearest Neighbor
- kNN
- SVM
- ...

Nearest Neighbor Classifier

• Train
  – Remember all training images and their labels

• Predict
  – Find the closest (most similar) training image
  – Predict its label as the true label
How to find the most similar training image? What is the distance metric?

L1 distance:

\[ d_1(I_1, I_2) = \sum_p |I^p_1 - I^p_2| \]

Where \( I_1 \) denotes image 1, and \( p \) denotes each pixel.

<table>
<thead>
<tr>
<th>Test image</th>
<th>Training image</th>
<th>Pixel-wise absolute value differences</th>
</tr>
</thead>
<tbody>
<tr>
<td>56 32 10 18</td>
<td>10 20 24 17</td>
<td>46 12 14 1</td>
</tr>
<tr>
<td>90 23 128 133</td>
<td>8 10 89 100</td>
<td>82 13 39 33</td>
</tr>
<tr>
<td>24 26 178 200</td>
<td>12 16 178 170</td>
<td>12 10 0 30</td>
</tr>
<tr>
<td>2 0 255 220</td>
<td>4 32 233 112</td>
<td>2 32 22 108</td>
</tr>
<tr>
<td><strong>456</strong></td>
<td><strong>456</strong></td>
<td><strong>456</strong></td>
</tr>
</tbody>
</table>

Slides from Andrej Karpathy and Fei-Fei Li
http://vision.stanford.edu/teaching/cs231n/
Choice of distance metric

- Hyperparameter

L1 (Manhattan) distance

\[ d_1(I_1, I_2) = \sum_p |I_1^p - I_2^p| \]

L2 (Euclidean) distance

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

- Two most commonly used special cases of p-norm

\[ ||x||_p = \left( |x_1|^p + \cdots + |x_n|^p \right)^{\frac{1}{p}} \quad p \geq 1, x \in \mathbb{R}^n \]
k-nearest neighbor

- Find the k closest points from training data
- Labels of the k points “vote” to classify
How to pick hyperparameters?

• Methodology
  – Train and test
  – Train, validate, test

• Train for original model
• Validate to find hyperparameters
• Test to understand generalizability
Validation

Validation data
use to tune hyperparameters
evaluate on test set ONCE at the end
Cross-validation

cycle through the choice of which fold is the validation fold, average results.

Slides from Andrej Karpathy and Fei-Fei Li
http://vision.stanford.edu/teaching/cs231n/
Example dataset: CIFAR-10
10 labels
50,000 training images
10,000 test images.
Visualization: L2 distance
Complexity and Storage

- N training images, M testing images
- Training: $O(1)$
- Testing: $O(MN)$

- Hmm
  - Normally need the opposite
  - Slow training (ok), fast testing (necessary)
Summary

• Data-driven: Train, validate, test
  – Need labeled data

• Classifier
  – Nearest neighbor, kNN (approximate NN, ANN)
Score function

class scores
Linear Classifier

Define a score function

\[ f(x_i, W, b) = Wx_i + b \]

data (image)

class scores

“weights”

“bias vector”

“parameters”
Linear Classifier

\[ f(x_i, W, b) = W x_i + b \]

(assume CIFAR-10 example so 32 x 32 x 3 images, 10 classes)
Computing scores

input image

stretch pixels into single column

\[
W = \begin{bmatrix}
0.2 & -0.5 & 0.1 & 2.0 \\
1.5 & 1.3 & 2.1 & 0.0 \\
0 & 0.25 & 0.2 & -0.3
\end{bmatrix}
\]

\[
b = \begin{bmatrix}
56 \\
231 \\
24 \\
2
\end{bmatrix}
\]

\[
f(x_i; W, b) = \begin{bmatrix}
-96.8 \\
437.9 \\
61.95
\end{bmatrix}
\]

cat score

dog score

ship score
Geometric Interpretation

\[ f(x_i, W, b) = W x_i + b \]
Interpretation: Template matching

\[ f(x_i, W, b) = W x_i + b \]
Linear classifiers

- Find linear function (hyperplane) to separate positive and negative examples

\[ x_i \text{ positive: } x_i \cdot w + b \geq 0 \]
\[ x_i \text{ negative: } x_i \cdot w + b < 0 \]

Which hyperplane is best?
Support vector machines

• Find hyperplane that maximizes the *margin* between the positive and negative examples

Support vector machines

• Find hyperplane that maximizes the *margin* between the positive and negative examples

\[ x_i \text{ positive } (y_i = 1) : \quad x_i \cdot w + b \geq 1 \]
\[ x_i \text{ negative } (y_i = -1) : \quad x_i \cdot w + b \leq -1 \]

For support, vectors, \[ x_i \cdot w + b = \pm 1 \]
Support vector machines

• Find hyperplane that maximizes the *margin* between the positive and negative examples

\[ x_i \text{ positive } (y_i = 1): \quad x_i \cdot w + b \geq 1 \]

\[ x_i \text{ negative } (y_i = -1): \quad x_i \cdot w + b \leq -1 \]

For support, vectors,

\[ x_i \cdot w + b = \pm 1 \]

Distance between point and hyperplane:

\[ \frac{|x_i \cdot w + b|}{||w||} \]

Therefore, the margin is

\[ 2 / ||w|| \]
Bias Trick

\[
\begin{bmatrix}
0.2 & -0.5 & 0.1 & 2.0 \\
1.5 & 1.3 & 2.1 & 0.0 \\
0 & 0.25 & 0.2 & -0.3 \\
\end{bmatrix}
+ \begin{bmatrix}
56 \\
231 \\
24 \\
2 \\
\end{bmatrix}
= \begin{bmatrix}
0.2 & -0.5 & 0.1 & 2.0 & 1.1 \\
1.5 & 1.3 & 2.1 & 0.0 & 3.2 \\
0 & 0.25 & 0.2 & -0.3 & -1.2 \\
\end{bmatrix}
\]

new, single \( W \)