Schedule

Today

- Histogram of Oriented Gradients (HOG)
- Dimensionality Reduction with Principal Component Analysis (PCA)
- Going Deeper into Classification
  - Underfitting / Overfitting
  - Training-Test-Validation
- Support Vector Machine (SVM)
- Multi-Class SVM
Classification pipeline

Feature Extraction

\[
\begin{bmatrix}
2 \\
5 \\
1 \\
8
\end{bmatrix}
\]

Classifier

\{\text{cat, dog}\}
Classification pipeline (Multi-class)

Feature Extraction
Histogram-of-Oriented-Gradients

\[
\begin{bmatrix}
2 \\
5 \\
1 \\
8 \\
\end{bmatrix}
\]

Multi-class SVM

\{speed limit 20, speed limit 30, ..., derestiction, yield way, ...\}
Histogram-of-Oriented-Gradients

- Dalal & Triggs 2005
- Initially used for pedestrian detection
- Describes local gradient orientation distribution
Histogram-of-Oriented-Gradients

- Dalal & Triggs 2005
- Initially used for pedestrian detection
- Describes local gradient orientation distribution
- Compute gradients
  - Convolute image with
    
    \[
    \begin{bmatrix}
    -1 & 0 & 1
    \end{bmatrix}
    \quad \text{and} \quad
    \begin{bmatrix}
    0 & -1 & 0
    \end{bmatrix}
    \]
  - Yields pixel-wise orientation
Histogram-of-Oriented-Gradients

- Dalal & Triggs 2005
- Initially used for pedestrian detection
- Describes local gradient orientation distribution
- Compute gradients
  - Convolute image with
    \[
    \begin{bmatrix}
    -1 & 0 & 1 \\
    \end{bmatrix}
    \quad \text{and} \quad
    \begin{bmatrix}
    -1 \\
    0 \\
    1 \\
    \end{bmatrix}
    \]
  - Yields pixel-wise orientation
- Divide image into cells (e.g., 8x8 pixels)
Histogram-of-Oriented-Gradients

- Dalal & Triggs 2005
- Initially used for pedestrian detection
- Describes local gradient orientation distribution
- Compute gradients
  - Convolute image with
    
    \[
    \begin{bmatrix}
    -1 & 0 & 1 \\
    \end{bmatrix}
    \quad \text{and} \quad
    \begin{bmatrix}
    -1 \\
    0 \\
    1 \\
    \end{bmatrix}
    \]
  - Yields pixel-wise orientation
- Divide image into cells (e.g., 8x8 pixels)
- Compute a histogram of all orientations present in each cell
  - Weigh the contribution of each pixel with its absolute gradient magnitude
Histogram-of-Oriented-Gradients

- Dalal & Triggs 2005
- Initially used for pedestrian detection
- Describes local gradient orientation distribution
- Compute gradients
  - Convolute image with 
    \[
    \begin{bmatrix}
    -1 & 0 & 1 \\
    \end{bmatrix}
    \quad \text{and} \quad 
    \begin{bmatrix}
    -1 \\
    0 \\
    1 \\
    \end{bmatrix}
    \]
  - Yields pixel-wise orientation
- Divide image into cells (e.g., 8x8 pixels)
- Compute a histogram of all orientations present in each cell
  - Weigh the contribution of each pixel with its absolute gradient magnitude
- Combine neighbouring cells to blocks (e.g. 2x2 cells) and normalize histograms with respect to sum of all pixel gradients magnitudes
Dalal & Triggs 2005
Initially used for pedestrian detection
Describes local gradient orientation distribution
Compute gradients
- Convolute image with
  \[
  \begin{bmatrix}
  -1 & 0 & 1 \\
  0 & 1 \\
  
  \end{bmatrix}
  \]
- Yields pixel-wise orientation
Divide image into cells (e.g., 8x8 pixels)
Compute a histogram of all orientations present in each cell
- Weigh the contribution of each pixel with its absolute gradient magnitude
Combine neighbouring cells to blocks (e.g. 2x2 cells) and normalize histograms with respect to sum of all pixel gradients magnitudes
For all blocks for all cells concatenate the histograms
Histogram-of-Oriented-Gradients
Visualizing High-dimensional Feature Spaces

- High-dimensional vectors are hard to interpret
- Visualizing in 2d or 3d is preferable
- Dimensionality reduction / embedding
- Several methods:
  - PCA (Principal Component Analysis)
  - t-SNE (t-distributed Stochastic Nearest-Neighbour Embedding)
  - LLE (Locally-Linear Embedding)
  - MDS (Multi-Dimensional Scaling)
Visualizing High-dimensional Feature Spaces
Visualizing High-dimensional Feature Spaces

- Find function that maps data points to 2 dimensions: \( f : \mathbb{R}^n \rightarrow \mathbb{R}^2 \)
- Make it easy: Linear
- Thus, can be represented by a 2 x n matrix
  \[
  f(x) = Lx = \begin{pmatrix} l_1 \\ l_2 \end{pmatrix} x
  \]
- But linear means 0 is mapped to 0
  - Subtract mean value from dataset beforehand
- Consists of two rows
- Rows represent the axes of main variance (principal axes)
Visualizing High-dimensional Feature Spaces

- Rows represent the axes of main variance (principal axes)

\[
\min_{l_1, \|l_1\| = 1} \sum_i (x_i^T l_1)^2
\]

- Row vector maximizing this, is given by eigenvector of

\[
C = \sum_i x_i x_i^T
\]

w.r.t. largest eigenvalue (covariance matrix \(C\))

- Generally: Take the eigenvectors corresponding to the largest eigenvalues of the covariance matrix and project the zero-mean dataset to these vectors
\[
\begin{align*}
&\min_{l_1, \|l_1\|^2 = 1} \sum_i (x_i^T l_1)^2 \\
= &\min_{l_1, \|l_1\|^2 = 1} \sum_i l_1^T x_i x_i^T l_1 \\
= &\min_{l_1} \max_{\lambda \geq 0} \sum_i l_1^T x_i x_i^T l_1 + \lambda(\|l_1\|^2 - 1) \\
\min_{\lambda \geq 0} &2 \sum_i x_i x_i^T l_1 + 2\lambda l_1 = 0 \\
\min_{\lambda \geq 0} &\left( \sum_i x_i x_i^T \right) l_1 = \lambda l_1
\end{align*}
\]
Image Classification

- Linear classifier finds hyperplane to separate sets of points
- A more complex classifier might find a better way to separate the two datasets
Image Classification

- Linear classifier finds hyperplane to separate sets of points
- A more complex classifier might find a better way to separate the two datasets
- Many ML methods have hyper-parameters that control the complexity of the function to fit
- But: In general, very complex functions tend to perform worse on unseen data
Image Classification

- Linear classifier finds hyperplane to separate sets of points
- A more complex classifier might find a better way to separate the two datasets
- Many ML methods have hyper-parameters that control the complexity of the function to fit
- But: In general, very complex functions tend to perform worse on unseen data
Image Classification

- Underfitting
- Good fit
- Overfitting

Error vs. Complexity

Test error vs. Training error

Image Classification Graph
INTRODUCTION TO DEEP LEARNING FOR COMPUTER VISION

Image Classification

Underfitting  Good fit  Overfitting

Training error  Test error

Error

Complexity

-2 -1 0 1 2 3 4 5 6
0 2 4 6 8 10
Image Classification

- Underfitting
- Good fit
- Overfitting

Error vs Complexity

Training error
Test error
**Image Classification**

- Linear classifier finds hyperplane to separate sets of points
- A more complex classifier might find a better way to separate the two datasets
- Many ML methods have hyper-parameters that control the complexity of the function to fit
- But: In general, very complex functions tend to perform worse on unseen data
- Need to estimate the training error: split dataset into training-validation-test
Support Vector Machines
Support Vector Machines

- Labelled Data: \((x_i, y_i), x_i \in \mathbb{R}^n, y_i \in \{-1, 1\}\)
- Solve:

\[
\min_{w,b} \quad w^T w
\]
\[
s.t. \quad y_i \cdot (w^T x_i + b) \geq 1
\]
Support Vector Machines

- Labelled Data: \((x_i, y_i), x_i \in \mathbb{R}^n, y_i \in \{-1, 1\}\)
- Solve:

\[
\min_{w, b} \quad w^T w + C \sum_i \xi_i \\
\text{s.t.} \quad y_i \cdot (w^T x_i + b) \geq 1 - \xi_i \\
\xi_i \geq 0
\]
Support Vector Machines

- Labelled Data: \((x_i, y_i), x_i \in \mathbb{R}^n, y_i \in \{-1, 1\}\)
- Solve:

\[
\min_{w, b} \quad w^T w + C \sum_i \xi_i \\
\text{s.t.} \quad y_i \cdot (w^T \phi(x_i) + b) \geq 1 - \xi_i \\
\xi_i \geq 0
\]

- \(C, \gamma\) are a hyper-parameter that control complexity
- Multiclass: One-vs-All
  - most confident classifier wins
  - Confidence ist given by distance to border
- Multiclass: One-vs-One
QUESTIONS?
EXERCISES.