COMP3204/COMP6223 - Computer Vision

2

Machine learning for Pattern Recognition

Summary

Machine learning is a fundamental part of high-level (interpreting what is seen, versus the low-level, which is all about processing the image) computer vision. The standard computer vision pipeline goes from the image, through a process of feature extraction, to a pattern recognition component that makes inferences. Machine learning is a standard way of training the pattern recognition system.

Key points

Feature spaces

- A feature space is an abstract mathematical space defined by a **feature extraction procedure** that transforms raw data into **featurevectors** of some **fixed number of elements**.
  - A featurevector is just a list of numbers that represents a point in the corresponding feature space.
    - The vector can also be considered to represent a direction in the space.
  - The number of elements of the vector is known as the **dimensionality** of the space (and the vector)
- Feature extraction is a fundamental part of computer vision.
  - Future lectures will cover some specific types of feature extraction in more detail.

Distance and similarity measures

- Feature extractors are often defined so that they produce vectors that are *close* together for *similar* inputs (for some given notion of similarity between the input objects)
  - Closeness of two vectors can be computed in the feature space by measuring a distance between the vectors.
  - The most common distance measure is the Euclidean Distance or L2 distance, which represents the straight-line distance between two points $\mathbf{p}=(p_1, p_2, \ldots, p_n)$ and $\mathbf{q}=(q_1, q_2, \ldots, q_n)$:

    $$\mathrm{d}(\mathbf{p},\mathbf{q}) = \mathrm{d}(\mathbf{q},\mathbf{p}) = \sqrt{(q_1-p_1)^2 + (q_2-p_2)^2 + \cdots + (q_n-p_n)^2} = \sqrt{\sum_{i=1}^{n}(q_i-p_i)^2}.$$

    and equivalently in vector form:
    $\|\mathbf{q}-\mathbf{p}\| = \sqrt{(\mathbf{q}-\mathbf{p})\cdot(\mathbf{q}-\mathbf{p})}$.
  - The L1 distance (aka taxicab or Manhattan distance) is also often used:

    $$d_1(\mathbf{p},\mathbf{q}) = \|\mathbf{p}-\mathbf{q}\|_1 = \sum_{i=1}^{n}|p_i-q_i|,$$

  - The Cosine similarity is another commonly used measure:

    $$\text{similarity} = \cos(\theta) = \frac{A\cdot B}{\|A\|\|B\|} = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n}(A_i)^2} \times \sqrt{\sum_{i=1}^{n}(B_i)^2}}$$

    - **This is not a distance measure!**
      - similarity=1 if vectors are in the same direction; decreases as angle increases
    - Useful if you only care about relative direction, but not magnitude

Written by Jonathon Hare. Send any comments/bugs/typos to jsh2@ecs.soton.ac.uk.

Supervised machine learning: classification

- Classification is the process of assigning a **class label** to an object (typically represented by a vector in a feature space).
    - A supervised machine-learning algorithm uses a set of **pre-labelled training data** to learn how to assign class labels to vectors (and the corresponding objects).
    - A **binary classifier** only has two classes; a **multiclass classifier** has many classes.
- A linear classifier tries to learn a **hyperplane** that separates the feature space in two (it's a binary classifier), such that all the training points of one class lie on one side of the plane, and all the points of the other class lie on the other side.
    - Practically speaking, there are an infinite number of possible hyperplanes. In the real world, the data might not be completely linearly separable, so algorithms for learning a hyperplane try to **minimise error** in some respect.
        - In the lecture, the linear-classifier demo used an algorithm called the **Perceptron learning algorithm** to learn the hyperplane.
    - Linear classifiers tend to be much more efficient than KNN once trained as they don't have to hold on to the training data; only a simple check of what side of the hyperplane the unlabelled point is on is required.
- Another common technique for learning a binary classifier is the Support Vector Machine (SVM).
    - A standard SVM is a linear classifier, just like the Perceptron (albeit producing a different hyperplane).
    - SVMs have the property that they can easily be made nonlinear, and create decision boundaries that are not flat planes.
- Intuitively, the simplest form of multiclass supervised classification technique is **K-nearest-neighbours (KNN).**
    - The class of an unlabelled vector is determined by finding the majority class of the *closest* (by some measure) *K* training points.
    - KNN can extend to any number of classes (it is multiclass), but becomes computationally **expensive** when there are many training examples, or the dimensionality of the feature space is large.

Unsupervised machine learning: clustering

- Clustering is an unsupervised machine learning technique, that learns to group data without prior knowledge of what the groups should look like.
    - The **K-Means** algorithm is a simple approach to clustering that attempts to group data in a feature space into K groups or clusters represented by centroids (i.e. the mean point of the class in feature-space).
        - The K-value must be set *a priori* (beforehand)
        - To begin, K initial cluster centres are chosen (typically randomly or from a sample of the existing data points)
        - Then the following process is performed iteratively until the centroids don't move between iterations (or the maximum number of iterations is reached):
            - Each point is assigned to its closest centroid
            - The centroid is recomputed as the mean of all the points assigned to it. If the centroid has no points assigned it is randomly re-initialised to a new point.
        - The final clusters are created by assigning all points to their nearest centroid.
    - **K-Means always converges, but not necessarily to the most optimal solution**
    - Other more advanced clustering algorithms don't need to know the number of clusters *a priori* (but you don't need to know the details for this course).

Further reading

- Mark's book (third edition) p424-429 covers K-nearest-neighbours and some other approaches.
- Wikipedia has good entries for:
  - KNN (http://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm)
  - K-Means (http://en.wikipedia.org/wiki/K-means_clustering).
  - Perceptron (http://en.wikipedia.org/wiki/Perceptron)
- Now would also be a good time to refresh your knowledge on vectors (in particular concepts such as dot product (or inner product) and length. The Wikipedia page on Euclidean vectors (http://en.wikipedia.org/wiki/Euclidean_vector) is a good starting point.

Practical exercises

- Chapter 3 of the OpenIMAJ tutorial covers k-means clustering along with some topics we'll cover in future lectures.
- If you feel like getting really advanced (don't worry if not, we'll revisit this in a later lecture), the following classes let you do supervised classification in OpenIMAJ (using KNN and a linear classifier) – you could try to use them to classify something:
  - `org.openimaj.ml.annotation.basic.KNNAnnotator`
  - `org.openimaj.ml.annotation.linear.LiblinearAnnotator`

Written by Jonathon Hare. Send any comments/bugs/typos to jsh2@ecs.soton.ac.uk.