Lecture 3
Introduction to Concept Learning

Topics of this lecture

- General considerations
  - Learn what -> concept learning
    - Learn relations and predict the future
  - Learn how -> procedural learning
    - Learn methods for solving given problems
- Basic knowledge for concept learning
  - Pattern classification and recognition
  - Feature vector representation of patterns
  - Nearest neighbor based learning
  - Discriminant function and decision boundary
  - Multi-class problem
  - The LVQ algorithm

Learn what -> concept learning

- There are **declarative knowledge** and **procedural knowledge**.
- The former consists of various "concepts" (e.g. apple, fruits, plants, animals, etc.) and the latter consists of "instructions" for doing something.
- Instructions, in turn, can be described by using sequences of concepts (like sentences of words) -> "**Concept learning**" is the "basic" for machine learning.
- Each concept corresponds to a certain class of "patterns". Thus, "concept learning" is actually "pattern learning", which is the foundation of **pattern recognition**.

Examples of concepts to learn

- Characters (kanji, kana, digits, etc.)
- Biometrics (finger prints, vein, iris, etc.)
- Signals (speech, sound, speaker, etc.)
- Images (face, expression, human, etc.)
- Videos (gaits, activity, gesture, etc.)
- Objects (human, car, obstacle, etc.)
- Anomaly (product, text, sequence, etc.)
- Time series (risk, chance, etc.)

Learn what

- A goal of learning is to learn properties of some concepts or relations between concepts, and predict the future.
- Events that may occur in a near future can be predicted if they are closely correlated with events that have already occurred.
  - Weather forecast: winter-type distribution pattern of atmospheric pressure (has certain property).
  - Stock market: golden cross (has hints to start a new cycle).
- Once we have designed a learning model or a hypothesis h based on existing patterns, h can predict patterns that may occur with a high probability.
  - Word imbedding: predict the next word or phrase given a context.
  - Generative model: produce new patterns via simulation.

Learn how

- The main objective of procedural learning is to learn a set of instructions for doing something.
  - To find the shortest route to the Inawashiro lake.
  - To find the simplest proof for a theorem.
  - To find the most efficient method for solving a problem.
- Each instruction is a **{situation, action}** pair, and a way for doing something is a sequence of instructions that maps a given situation to an action.
- In principle, procedural learning can be decomposed to several "situation (pattern) recognition" problems.
- Usually, the situation recognizer is embedded in a search tree or graph (e.g. Alpha-Go).

Learn methods for solving problems

- Problem solving is one of the main goals of AI research.
- Given any problem, it is expected that AI can find the solution and solve the problem without being programmed.
- *To find a good way for solving a problem* is more difficult than *to find the solution* itself.
- Learning of problem solving is procedural learning.
- The solution may contain many steps, and each step contains some instructions for solving a sub-problem.

Definition of a concept

- Concept is a sub-set of the **universe of discourse**.
- X: Universe of discourse
- A: concept defined on X

$$A=\{x \text{ in X} \mid \mu_A(x)=\text{True}\}$$

- where $\mu_A(x)$ is a logic formula representing the **membership function** of A.
- For a fuzzy concept, the range of $\mu_A(x)$ is [0,1].

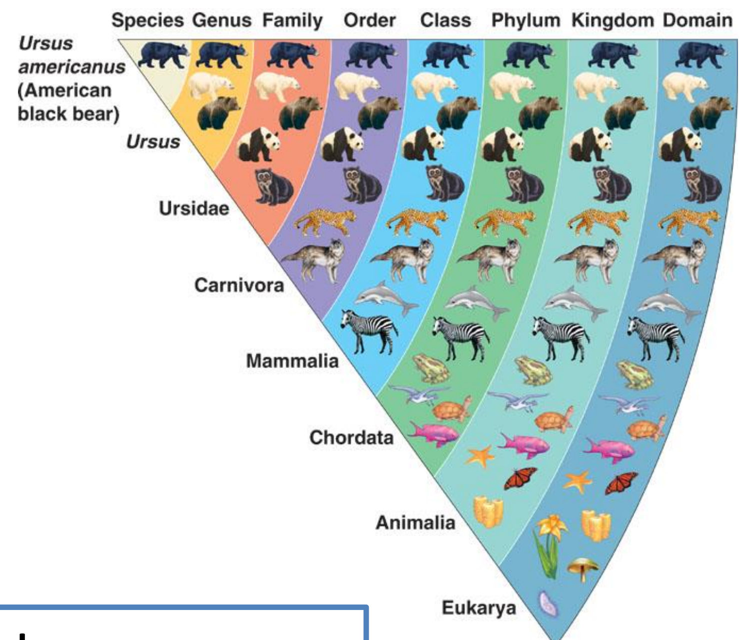
Pattern classification and recognition

- Pattern classification is the process for partitioning the universe of discourse X into various meaningful concepts.
- Pattern recognition is the process to determine to which concept an observed datum belongs.
- Usually recognition and classification are considered synonyms in the literature.
- Example:
  - Domain: Chinese characters (Kanji)
  - Concepts: Nouns, verbs, adjectives, …
  - Given observation: 城 → noun; 走 → verb; 良 → adjective
- A concept is also called a class, a category, a group, a cluster, etc. , depends on applications.

Why science is translated to “科学” (Kagaku, Ke-shue)?

- 「科学」is an interesting translation of the word “science”.
- It means “study on classification or categorization of objects or concepts” .
- Based on the classification results, we can understand the world in a more organized way, or “scientific way”.

https://liorpachter.wordpress.com/2015/10/27/straining-metagenomics/



Using categorized procedural knowledge, we can also solve problems “scientifically”.

Vector representation of patterns

- To classify or recognize objects in a computer, it is necessary to represent them numerically.
- We usually transform an object into an n-dimensional vector, which is a point in the n-D Euclidean space, as follows:

$$x = (x_1, x_2, \dots, x_n)^t$$

- Each element of the vector is called a **feature**, and the vector itself is called a **feature vector**. The set of all feature vectors is called the **feature space**.

Terminologies for learning

- Learning or training is the process for determining the membership functions of the concepts.
- **Training set** is a set of data used for learning. Each datum is called a training datum or **training pattern**.
- Usually, each training pattern x has a label, which tells the name of the concept x belongs to. The label is also called **teacher signal**.

Terminologies for learning

- In many applications, we consider two-class (binary or dichotomy) problems.
  - Face or non-face; Human or non-human; Normal or abnormal.
  - For two-class problems, the label takes only two values: positive or negative (1 or -1).
  - A pattern is called positive / negative pattern if its label is positive / negative.
- For any pattern, we can define its **neighborhood** using a distance measure. Usually, we use Euclidian distance. A pattern q is said close to x if the following distance is small.

$$\|\mathbf{x}-\mathbf{q}\| = \sqrt{\sum_{j=1}^{n}(x_j - q_j)^2}$$

Learning based on the neighborhood

- The simplest method for pattern classification is **nearest neighbor classifier** (NNC).
- To design an NNC, we just collect a set Ω of **labeled training data**, and use Ω directly for recognition.
- For any given pattern x, Label(x)=Label(p) if

$$p = arg \min_{q \in \Omega} |x - q|$$

- In this case, the whole training set Ω is used as an NNC.
- In general, NNC is defined by a **set P of prototypes** that can be a sub-set of Ω, or a set of templates *learned* from Ω.

Learning based on the neighborhood

- Using NNC, we can define the membership function of a concept A as follows:

$$\mu_A(\mathbf{x}) = [\exists \mathbf{p} \in P^+][\forall \mathbf{q} \in P^-] \|\mathbf{x} - \mathbf{p}\| \le \|\mathbf{x} - \mathbf{q}\|$$

- Where P^+ and P^- are the set of positive prototypes and set of negative prototypes, respectively.
- Physical meaning: For any given pattern x, if there is a positive prototype p, and the distance between x and p is smaller than that between x and any of the negative prototype, x belongs to A.

Properties of the NNC

- If the set P of prototypes contains enough number of observations, the error of the NNC is smaller than 2E, where E is the error of the “optimal” classifier (i.e. the Bayes error rate).
- However, if the size of P is too big, it is very time consuming to make a decision for any given pattern x.
- In other word, NNC is easy to obtain, but difficult to use.

A method for reducing the cost

- One method for reducing the computational cost is to use a **representative** for each class.
- For a 2-class problem, the representatives can be given by

$$\mathbf{r}^{+}=\frac{1}{|\Omega^{+}|}\sum_{\mathbf{p}\in\Omega^{+}}\mathbf{p},\quad \mathbf{r}^{-}=\frac{1}{|\Omega^{-}|}\sum_{\mathbf{q}\in\Omega^{-}}\mathbf{q},$$

where Ω^{+} and Ω^{-} are, respectively, the set of positive training data and set of negative training data.

- Use the representatives, recognition is conducted by

$$\mathrm{Label}(\mathbf{x})=\begin{cases}+1 & \text{if } \|\mathbf{x}-\mathbf{r}^{+}\|<\|\mathbf{x}-\mathbf{r}^{-}\| \\ -1 & \text{if } \|\mathbf{x}-\mathbf{r}^{-}\|<\|\mathbf{x}-\mathbf{r}^{+}\|\end{cases}$$

From NNC to discriminant functions

- If the distance is defined as the Euclidean distance, pattern recognition can also be conducted as follows:

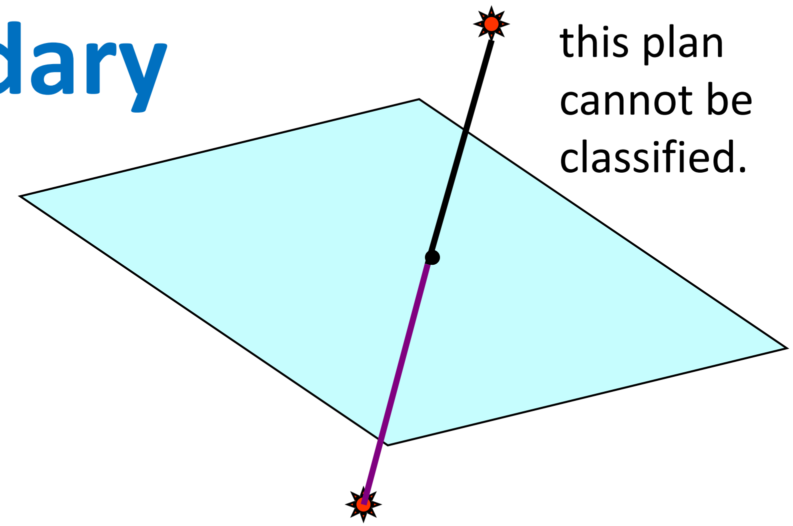
$$\text{Label}(\mathbf{x}) = \begin{cases} +1 & \text{if } g^{+}(\mathbf{x}) > g^{-}(\mathbf{x}) \\ -1 & \text{if } g^{+}(\mathbf{x}) < g^{-}(\mathbf{x}) \end{cases}$$

- Here, $g^{+}(x)$ and $g^{-}(x)$ are called discriminant functions defined by

$$g^{+}(\mathbf{x}) = \sum_{j=1}^{n} x_j r_j^{+} - \frac{1}{2}\sum_{j=1}^{n}(r_j^{+})^2, \quad g^{-}(\mathbf{x}) = \sum_{j=1}^{n} x_j r_j^{-} - \frac{1}{2}\sum_{j=1}^{n}(r_j^{-})^2$$

- Since both functions are linear, they are also called **linear discriminant functions**.

Linear decision boundary



- For a 2-class problem, we need only one discriminant function defined by

$$g(\mathbf{x}) = g^{+}(\mathbf{x}) - g^{-}(\mathbf{x}) = \sum_{j=1}^{n} w_j x_j - \theta$$

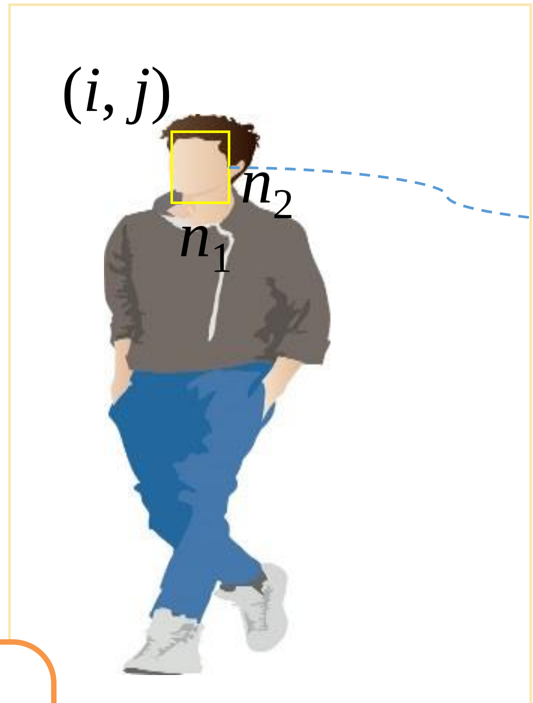
- This function is actually a **hyper-plane**. Thus, this hyper-plane forms the **decision boundary**.

$$H : \sum_{i=1}^{n} w_i x_i - \theta = 0$$

$$w_i = r_i^{+} - r_i^{-};$$

$$\theta = \frac{1}{2}\sum_{i=1}^{n}[(r_i^{+})^2 - (r_i^{-})^2]$$

Example: face detection



$\mathbf{x}=(x_{11},x_{12},\ldots,x_{n1,n2})^T$

Face Detector

+1 or -1

Any idea to locate the face efficiently?

NNC for face detection

- Collect positive examples -> Ω^+
  - Segment faces from given images / photos.
- Collect negative examples -> Ω^-
  - Segment sub-images that are not faces.
- Define the feature vectors
  - Reshape the segmented (n1xn2) images to n=n1xn2 dimensional feature vectors.
  - Reduce the dimension from n to m (<<n) using principal component analysis (PCA).
  - Each datum is represented using its projections to the m largest eigenvectors (also called eigen-faces).
  - All given examples together form an NNC.

Multi-class problem

- To solve a multi-class problem, we can use the following rule:

$$Given\ x, Label(x) = k\ if$$
$$k = arg\ \max_i g_i(x)$$

where $g_i(x)$ is the discriminant function of the i-th class defined by

$$g_i(x) = \sum_{j=1}^{n} x_j r_j^i - \frac{1}{2}\sum_{j=1}^{n}(r_j^i)^2, i = 1,2, \dots, N_c$$

and r^i is the representative of the i-th class.

Question: How **to find the representatives**, or how to find the discriminant functions directly?

Learning vector quantization (LVQ): An algorithm for finding the representatives

Step 1: Initialize the set P of all representatives at random.

Step 2: Take a training pattern x from Ω, and find the nearest neighbor p from P.

Step 3: If p has the same label as x, update it using Eq. (1); otherwise, update using Eq. (2)

$$p = p + \alpha(x - p) \quad (1)$$
$$p = p - \alpha(x - p) \quad (2)$$

Step 4: Step if terminating condition satisfied. Otherwise, return to Step 2.

Homework

- Try to provide another two-class pattern recognition problem, and describe briefly the process for solving the problem.