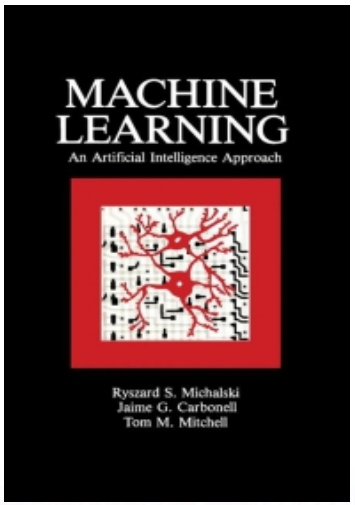
Different Conceptions of Learning

Function Approximation vs. Self-Organization

Pei Wang & Xiang Li
Temple University

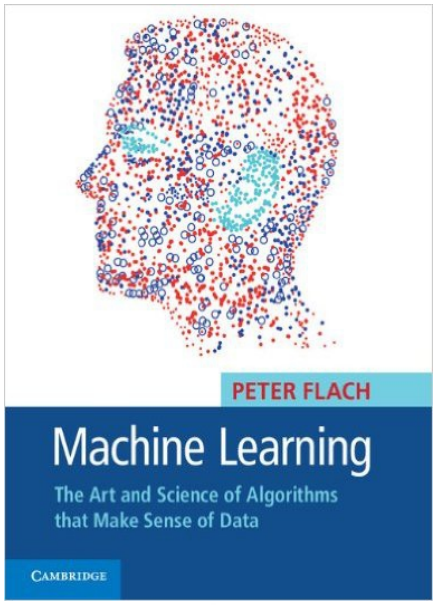
Many faces of learning

- Learning takes many forms, as known in psychology and other fields for decades
- There were also many competing paradigms in machine learning study in the early years



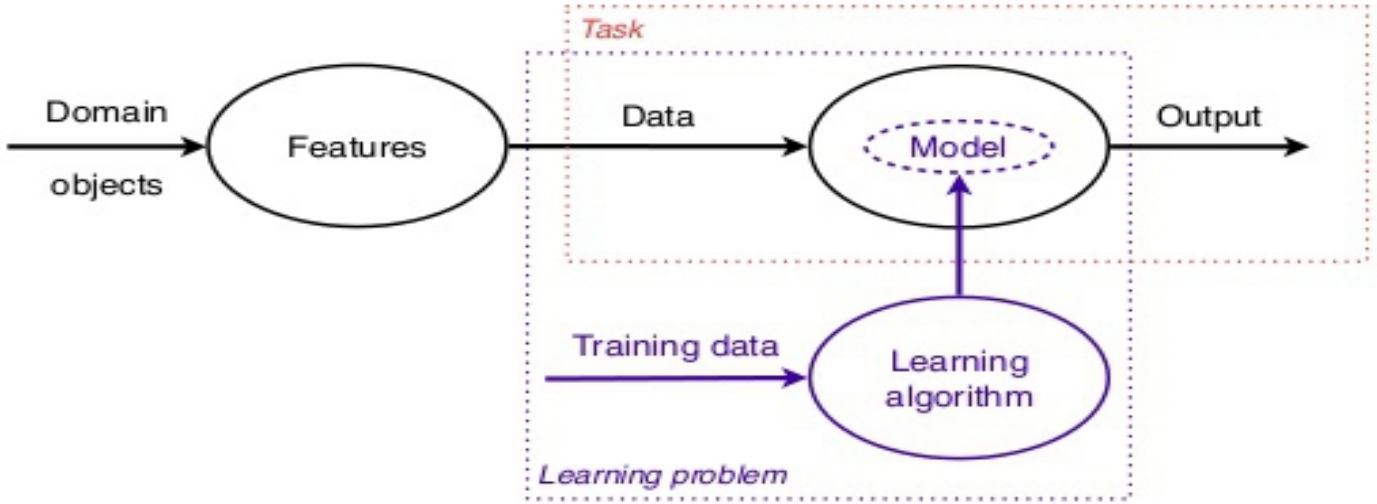
Algorithmic Learning

- In recent years, machine learning has been dominated by “algorithmic learning”
- “Using an algorithm to learn an algorithm”



Learning as meta-computation

Figure 3, p.11 How machine learning helps to solve a task



An overview of how machine learning is used to address a given task. A task (red box) requires an appropriate mapping – a model – from data described by features to outputs. Obtaining such a mapping from training data is what constitutes a learning problem (blue box).

Function approximation

- The algorithm to be learned serves as a function mapping input into output
- Training data are sample instances of the function
- “Learning” means generalization from known instances to unknown instances
- Assumption: similar inputs produce similar outputs in this approximation

Inferential learning

- Knowledge is represented as statements
- Learning is carried out by inference, especially non-deductive types, such as induction, abduction, analogy, ...
- This type of learning has also been explored in machine learning
- NARS will be introduced from this perspective

NARS overview

- NARS (Non-Axiomatic Reasoning System) is an adaptive system designed under the Assumption of Insufficient Knowledge and Resources (AIKR), which means finite, open, and real-time
- NARS uses a formal language for representation and formal rules for inference, and treats various cognitive functions as reasoning

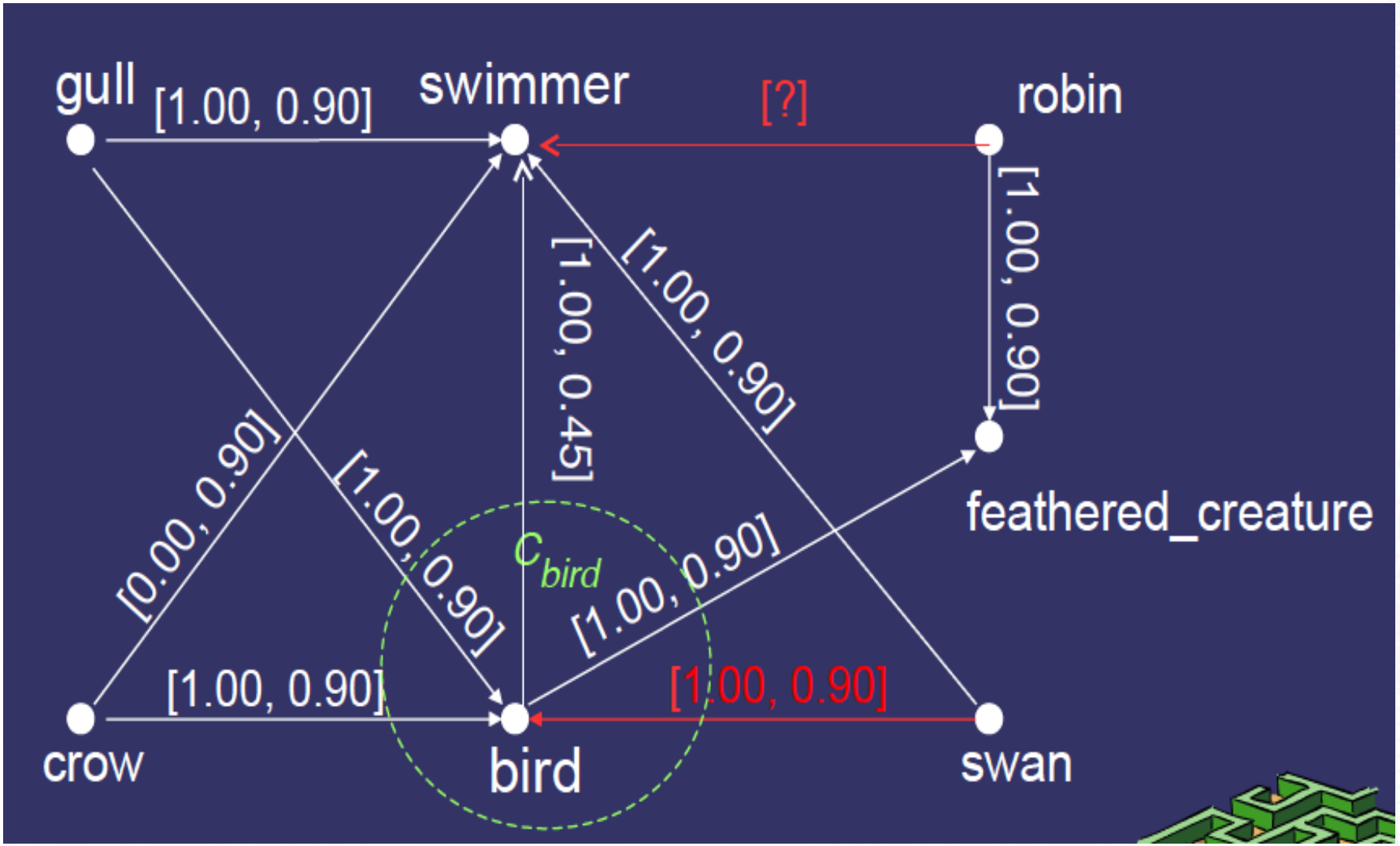
Knowledge in NARS

- Built-in knowledge: the grammar rules, inference rules, and control algorithms are hard-wired and domain-independent
- Empirical knowledge: all domain-specific knowledge comes from experience and is self-organized via learning
- Preloaded knowledge: “compiled” and “compressed” experience

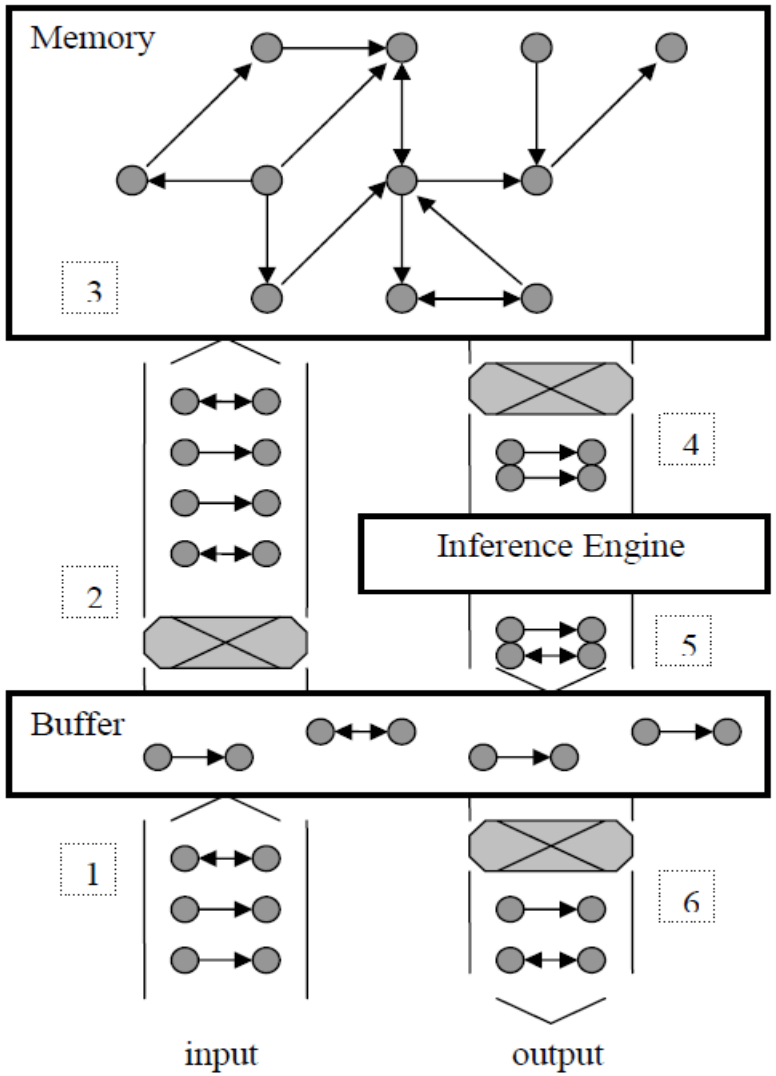
Empirical knowledge

- A *term* is an internal identifier of a *concept*, and represents an item or pattern in the system's experience
- A *belief* is a conceptual relationship with a measurement of evidential support
- A *task* can be a piece of experience to be absorbed, a question to be answered, or a goal to be achieved

Memory as a network



Architecture and routine



1. Input tasks are added into the task buffer.

2. Selected tasks are inserted into the memory.

3. Inserted tasks in memory may also produce beliefs and concepts, as well as change existing ones.

4. In each working cycle, a task and a belief are selected from a concept, and feed to the inference engine as premises.

5. The conclusions derived from the premises by applicable rules are added into the buffer as derived tasks.

6. Selected derived tasks are reported as output tasks.

Learning in NARS

- Input tasks may create new terms (and concepts) and beliefs
- Inference rules produce derived tasks, which may create new terms and beliefs
- Inference activities adjust priority values of concepts, tasks, and beliefs, so cause priming and forgetting, and change the meaning of concepts

Properties of learning

- Reasoning and learning are two aspects of the same process: single-step vs. lasting-effect
- All expressible empirical knowledge can be derived from proper experience
- Naturally *one-shot, incremental, anytime, online, life-long, active, transferable, multi-tasking, multi-strategy*, ...

Beyond algorithmic learning

- No separate learning and working phases
- The dynamic interaction of many micro-algorithms is not equivalent to an overall algorithm for task processing
- The input-output relation is not, and does not converge to, a fixed mapping
- This type of learning is more general and flexible, but less predictable