Extending ACT-R’s Modeling Capabilities: One Level Below, and One Level Above

Niels Taatgen
Bernoulli Institute, Artificial Intelligence
University of Groningen
Clusters of neurons (Nengo)
Representation: activation patterns

V1 = RT1
RT2 -> AC1

Primitive operations
Comparing and projecting patterns between workspace locations

Associated Learning
Pattern similarity

Skills
Sequences of operators with variable binding, processed serially

Operators
Sequences of primitive operations, processed in parallel

Composition
Reasoning, Instance-based learning

Tasks
Combinations of goals (parallel, sequential or hierarchical)

Reinforcement Learning
Reuse of Operators

Overlap in subsequences of operations

Conditions
Actions
Higher-level representations (language)
Lower-level neuronal representations

Neuromorphic architecture (Learning Materials)
Example: RITL experiment

Task 1
Instructions
SAME
SWEET
LEFT INDEX
3s
Trials
Answer: TRUE
(Left index finger)
Grape
Apple
Task 1 description:
If the answer to 'is it SWEET?'
is the SAME for both words,
press your LEFT INDEX finger

Task 2
Instructions
SECOND
LOUD
RIGHT MIDDLE
Trials
Answer: TRUE
(Right middle finger)
Leaf
Dynamite
Task 2 description:
If the answer to 'is it LOUD?'
is yes for the second word,
press your RIGHT MIDDLE finger
Just One
Loud
Left Middle
artificial intelligence + cognitive modeling
Drums
Airplanes
What skills do we need?

- Skills for same/just one/second/not second
- A skill that tests an attribute that we can instantiate
- A skill that pushes buttons that we can instantiate
Skill level model

- Primitives at the task level are skills
- Learning by a (linguistic) composition
- One-shot-learning!
We can compose these using chunks

- just-one-1
  - slot1 just-one
  - arg-subskill-1 determine-attribute-1
  - success-skill press-finger-yes
  - fail-skill press-finger-no

- determine-attribute-1
  - slot1 determine-attribute
  - fact-type loud

- press-finger-yes
  - slot1 press-finger
  - finger left-index

- press-finger-no
  - slot1 press-finger
  - finger right-index
The complete model

- Implementation of component skills: same, just-one, etc.
- A skill that builds the declarative structure that serves as a goal representation
Cole et al. 2017 experiment

- Limited time to study instruction (1100, 1900, 2700ms)
- Response within 1500ms
- Practiced vs. novel instructions

![Graph showing accuracy over CTI](image)
Model challenges

- How to get the model fast enough to be able to do this at all within the limited time available?
- How to model the difference between trained and novel?
Instruction parsing needs to generate four chunks

just-one-1
  slot1 just-one
  arg-subskill-1
determine-attribute-1
success-skill
  press-finger-yes
press-finger-yes
  slot1 press-finger
  finger left-index

fail-skill press-finger-no
press-finger-no
  slot1 press-finger
  finger right-index

determine-attribute-1
  slot1 determine-attribute
fact-type loud
Retrieval?

- If the instruction is “just-one loud left-index” again, you cannot just retrieve that.
- But we can retrieve based on one (just-one), and hope for the best.
- If we retrieve something else, we will modify it to fix it.
Procedure

- Try to retrieve based on the logical cue
  - If this fails, build a new chunk
- For the subskill, success-skill and fail-skill
  - First check whether the skill that is already there is correct
  - If not, try to retrieve the correct skill to replace it
  - If that fails, build it from scratch
Results
More general agenda

- Can we start building a general purpose skill library that can serve as a basis for more constrained modeling?
- If we have that, we can build instantiations of the cognitive architecture that already have a rich skill set
Clusters of neurons (Nengo)
Representation: activation patterns

V1 = RT1
RT2 -> AC1

Primitive operations
Comparing and projecting patterns between workspace locations

Operators
Sequences of primitive operations, processed in parallel

Skills
Sequences of operators with variable binding, processed serially

Composition
Reuse of Operators

Tasks
Combinations of goals (parallel, sequential or hierarchical)

Reasoning, Instance-based learning

Reinforcement Learning

Overlap in subsequences of operations

Conditions
Actions

Higher-level representations (language)

Lower-level neuronal representations

Neuromorphic architecture (Learning Materials)
Lower level of abstraction

- How can we learn operators from primitive operations?
  - Construct basic operators with a single primitive operation
  - Use “operator compilation” and reinforcement learning to discover the knowledge needed for a task
Example: Choice-reaction task

define facts {
    (f1 associate vanilla thumb)
    (f2 associate ice ring)
    (f3 associate paper pinkie)
    (fa1 taste vanilla sweet)
    (fa2 temperature ice cold)
    (fa3 weight paper small)
    (fb1 color vanilla yellow)
    (fb2 texture ice slippery)
    (fb3 color paper white)
}

Example basic operators

operator V1toRT2 {
    V1 <> nil
    ==> V1 -> RT2
    nil -> V1
}

operator RT2toRT1 {
    RT2 <> nil
    ==> RT2 -> RT1
}

operator RT1equalC1 {
    RT1 = *fact-type
    ==> }

Initial model
After learning
One possible solution

Operator 1
V1 <> nil
==> V1 - RT2
associate -> RT1

Operator 2
RT1 = associate
RT3 <> nil
==> action -> AC1
RT3 -> AC2
More complex task

define facts {
    (fac1 category goat animal)
    (fac2 category pinguin animal)
    (fac3 category cabbage plant)
    (fac4 category tulip plant)
    (altfac1 property goat hair)
    (altfac2 property pinguin notfly)
    (altfac3 property cabbage food)
    (altfac4 property tulip mania)
    (fca1 response animal left)
    (fca2 response plant right)
    (fca3 response somethingelse middle)
    (fca4 response differentlyet upper)
    (fca5 response noguessing lower)
    (altfca1 property animal livingthing)
    (altfca2 property plant livingthing)
}
artificial intelligence
cognitive modeling
Learning is more successful with prior learning.

- Two CRT tasks
- One CRT task
- No prior learning
Clusters of neurons
Representation: activation patterns
\( V_1 = R \)
\( T_1 \)
\( R \)
\( T_2 \rightarrow AC_1 \)

Primitive operations
Comparing and projecting patterns between workspace locations

Skills
Sequences of operators with variable binding, processed serially

Reasoning, Instance-based learning

Composition

Tasks
Combinations of goals (parallel, sequential or hierarchical)

Composition

Overlap in subsequences of operations

Reinforcement Learning

Reuse of Operators

Operators
Sequences of primitive operations, processed in parallel

Overlap in

Composition

Mark Ji:
Infant learning of simple grammatical patterns

Composition

Myself:
Implementation of PRIMs in Nengo

Conditions
Actions

Composition

Higher-level representations
(language)

Neuromorphic architecture (Learning Materials)

Composition

Composition

Composition

Composition

Composition

Composition

Composition

Composition