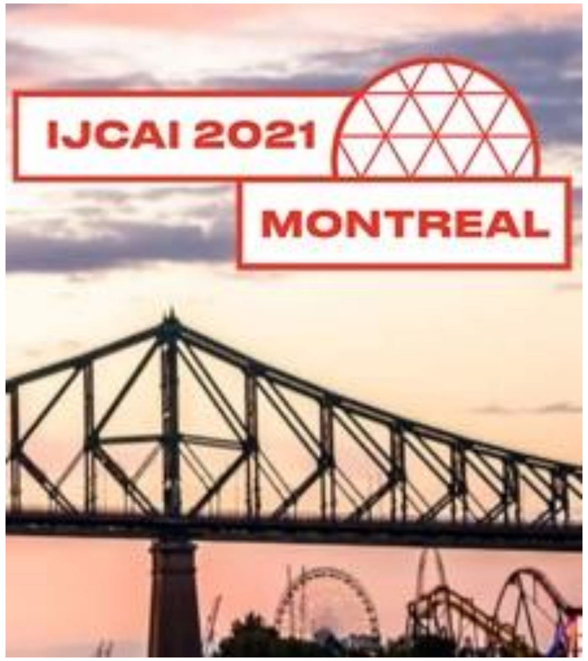
Knowledge-Based Translation of Natural Language into Symbolic Form

Christodoulos Ioannou and Loizos Michael

Computational Cognition Lab

LaCATODA

OPEN UNIVERSITY OF CYPRUS



Context of research

- Consider machines that have a knowledge for a domain (object-level knowledge) as a policy (i.e. cognitive assistants)
- Object-level knowledge cannot be always pre-programmed, instead it should be acquired by some means
- User should be able to amend machine policy, thus communication should be as natural as possible

Knowledge Acquisition

- Knowledge, especially common-sense knowledge, are hard to pre-program into a machine
- More realistically, it can be acquired through some form of learning, like machine learning
- In our context of research we need a knowledge acquisition method which incorporates some sort of advice-taking from a human. Such a method is Machine Coaching.
- To fulfil the natural communication requirement humans should, ideally, be able to advise machines using natural language

Machine Coaching protocol

- A cognitive assistant observes the state of the world and reasons using its current object-level knowledge on how to act
- A human coach observes the assistant and can inquire about the assistant's choices and offer advice back to the assistant on how to improve its knowledge
- Interaction between human and machine in this protocol can be done directly in symbolic form or by natural language or any other means of communication

A Machine Coaching Example (of object-level knowledge)

1. **Event**: Incoming call from some caller.
2. **User**: Declines. “Decline calls when busy.”
3. **Cognitive Call Assistant (CCA)**: “When do you consider yourself to be busy?”
4. **User**: Uses calendar and map applications and perhaps verbal commands to instruct that she considers herself busy when being in a meeting at work.
5. **Event**: (Later in the week...) Incoming call from John while user is in a work meeting.
6. **CCA**: Declines the call.
7. **User**: “Why did you decline this call?”
8. **CCA**: “Because you are at work and at a meeting, so I conclude that you are busy.”
9. **User**: “Send SMS saying `Busy! Will call back later.' to the caller when the call is important. If John is the caller then the call is important.”

Translating natural language into symbolic form

- How can natural language advice be translated into some symbolic form that can be used for reasoning by the assistant?
- This is a well-studied problem and numerous relevant parsing tools are available using, ontologies or other scope-related information, Controlled Natural Languages (CNL) and machine learning
- To our knowledge, systems built upon such parsing tools are effectively pre-programming the translation meta-level knowledge (of how to interpret object-level advice)
- Is there an alternative to pre-programming feasible and effective?
- Can this meta-level knowledge be learned?
- Can this meta-level knowledge also be acquired by means analogous to how the object-level knowledge is acquired, i.e. Machine Coaching?

Knowledge-based translation

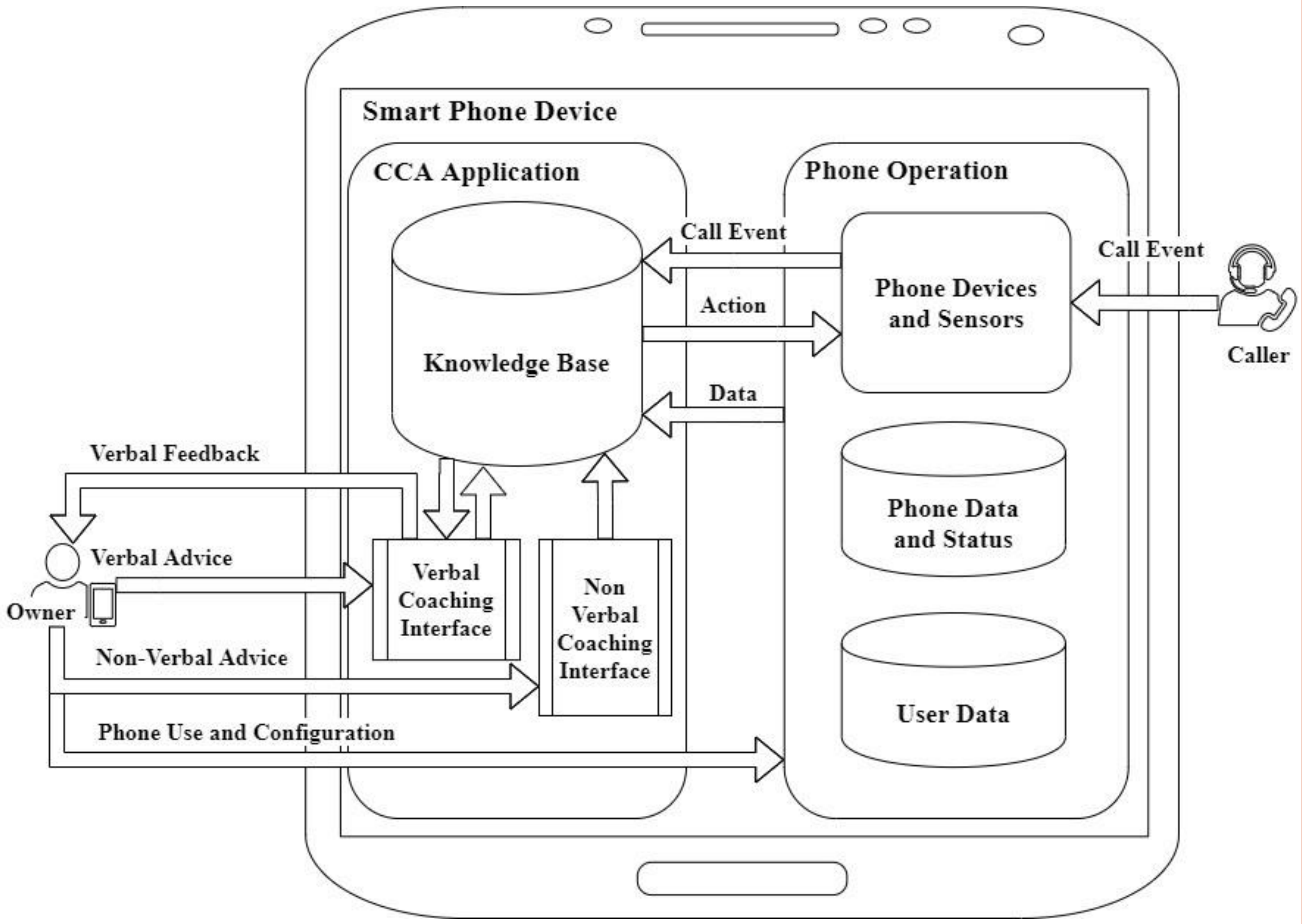
- We take a first step towards answering these questions by demonstrating how a translation meta-level knowledge can be engineered
- Although this amounts to pre-programming, it demonstrates a key prerequisite:

Meta-level knowledge is expressed and reasoned with, under the same syntax and semantics as the object-level knowledge, which is amenable to acquisition through Machine Coaching

A Machine Coaching Example (extended to meta-level knowledge)

Suppose that CCA fails to translate the last advice sentence from the user in the previous example:

10. **CCA**: “I am unable to interpret the advice `If John is the caller then the call is important’. Do you want to consult an expert to help improve the translation policy?”
11. **User**: “Yes, please!”
12. **CCA**: “Please provide additional information that you wish to pass on to the expert.”
13. **User**: “By this sentence I mean that I wish all calls coming from John to be labeled as important. This, in turn, will trigger a different treatment by the CCA.”
14. **Event**: The advice that could not be interpreted, along with the additional information provided by the user are sent to an expert. The expert machine coaches the CCA to improve its meta-level knowledge.
15. **CCA**: “Expert advice has been received and the translation policy has been amended. Please continue!”



Cognitive Call Assistant (CCA) Architecture

Supported types of sentences

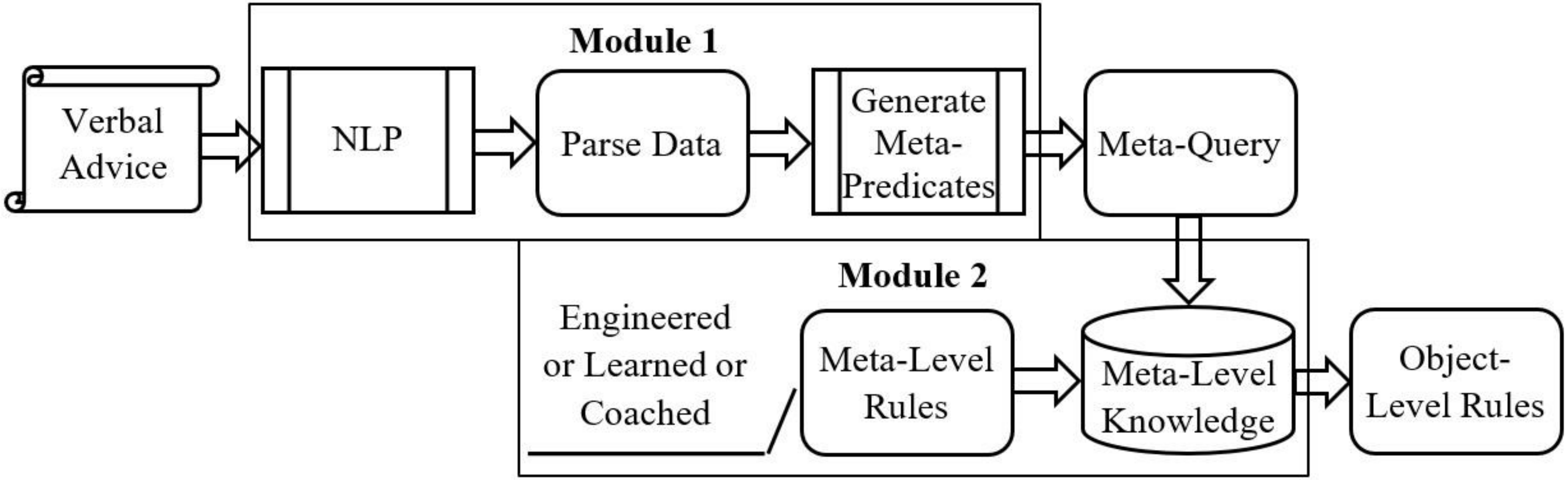
1. Simple Declarative Sentences (SDS)
    - 1.1. Simple Declarative To-Be Sentences
        - 1.1.1. with Adjective or Noun Predicate
        - 1.1.2. with Verb Predicate
    - 1.2. Simple Declarative Verb Sentences
        - 1.2.1. with Object
        - 1.2.2. without Object

2. Simple Imperative Sentences (SIS)

3. Conditional Sentences

Below SDC stands for Simple Declarative Clause and SIC stands for Simple Imperative Clause.

- 3.1. Imperative Conditional Sentences (ICS)
    - 3.1.1. If/When SDC, SIC
    - 3.1.2. If SDC, then SIC
    - 3.1.3. SIC if/when SDC
    - 3.1.4. all above with Implied Subject in SDC
- 3.2. Declarative Conditional Sentences (DCS)
    - 3.2.1. If/When SDC, SDC
    - 3.2.2. If SDC, then SDC
    - 3.2.3 SDC if/when SDC
    - 3.2.4. all above with Implied Subject in SDC



Implementation pipeline for a Verbal Coaching Interface

Syntax of the Meta-Level Knowledge

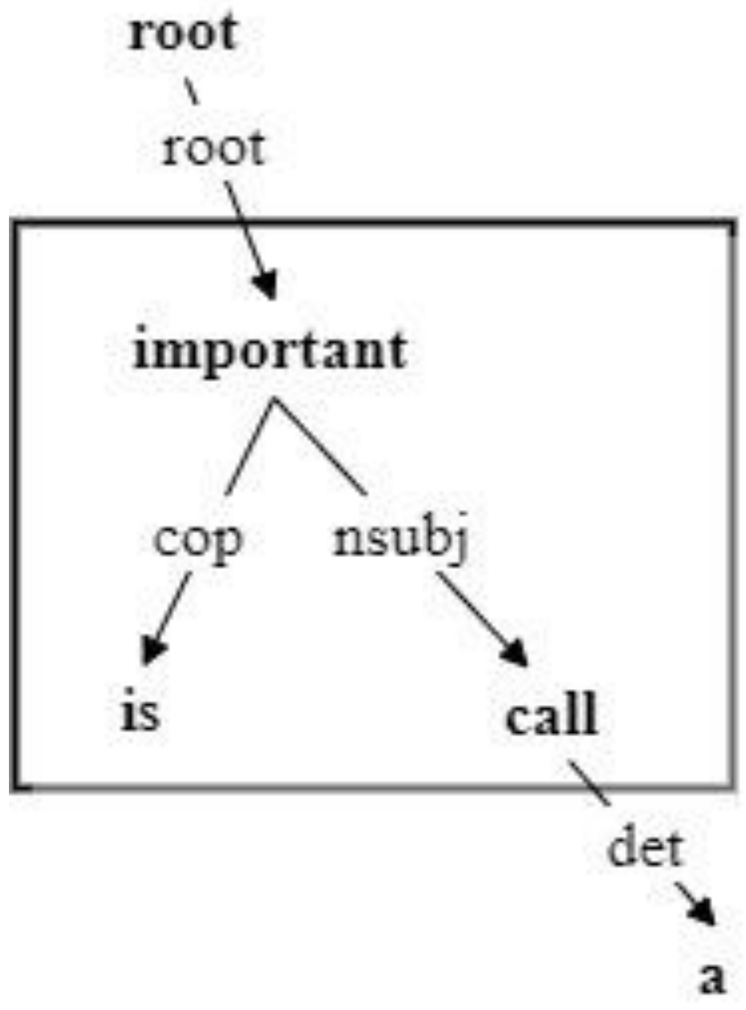
$$< \boldsymbol{Dependency} > (\quad ParentWord, ParentWordPosition,$$

$$ChildWord, ChildWordPosition)$$

$$\boldsymbol{pos}(< POSTag >, Word, WordPosition)$$

$$\boldsymbol{ner}(< NERTag >, Word, WordPosition)$$

$$\boldsymbol{ruleterms}(HeadTerms, HeadVars, BodyTerms, BodyVars)$$



Example Sentence 1: *A call is important.*

Type: *1.1.1. Simple Declarative To-Be Sentence with Adjective Predicate.*

Expected Rule: $call(X) implies\ important(X);$

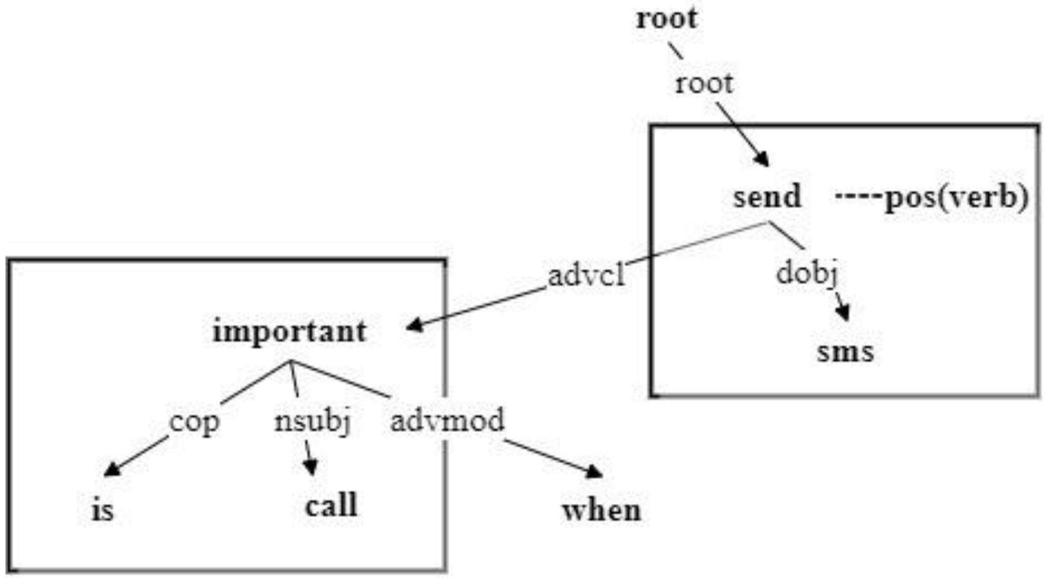
Applicable Meta-Rules:

$root(root, 0, W_1, P_1), cop(W_1, P_1, be, \_), nsubj(W_1, P_1, W_2, \_)$
$implies\ rulterms([[W_1]], [[X1]], [[W_2]], [[X1]]);$

Meta-Inferences: $rulterms([[important]], [[X1]], [[call]], [[X1]]);$

Generated Rule: $call(X1) implies\ important(X1);$

Engineering the Meta-Level Rules



Example Sentence 2: *Send SMS when call is important.*

Type: *3.1.3. Imperative Conditional Sentence (SIC when SDC).*

Expected Rule: $call(X),\ important(X),\ SMS(Y)\ implies\ send(Y);$

Applicable Meta-Rules:

$advcl(\_,\_,W_1,P_1), cop(W_1,P_1,be,\_), nsubj(W_1,P_1,W_2,\_)$
$implies\ rulterms([[\quad]],[[\quad]],[[W_2],[W1]],[[X1],[X1]]);$

$root(root,0,W_1,P_1), pos(verb,W_1,P_1), dobj(W_1,P_1,W_2,\_)$
$implies\ rulterms([[W_1]],[[X2]],[[W_2]],[[X2]]);$

Meta-Inferences:

$rulterms([[\quad]],[[\quad]],[[call],[important]],[[X1],[X1]]);$
$rulterms([[send]],[[X2]],[[SMS]],[[X2]]);$

Generated Rule: $call(X1), important(X1), SMS(X2)\ implies\ send(X2);$

Engineering the Meta-Level Rules

Why knowledge-based translation?

- Such a translation process would support:
  - Natural languages dialects
  - Individual human particularities
  - Omitted or implied meanings (which differentiate between humans)
  - Unsupported language parts.
  - Dynamic and cognitively-light amendment procedure
  - Less cognitively-heavy, batch-mode, offline pre-programming

Preliminary Empirical Evaluation

- Implemented the abovementioned pipeline using:
  - Stanford CoreNLP Library for natural language parsing
  - SWI-Prolog for for quick prototyping the meta-level knowledge
  - Java Prolog Library (JPL) to interface with the meta-level knowledge
- This position paper does not provide a full-fleshed quantitative empirical evaluation

Qualitative evaluation of pipeline implementation outcome

- The generated object-level rules reasonably capture the meaning of their associated natural language sentences
- In some cases, slightly altering the syntax of a sentence, alters the produced parse data
- Simple sentences with omitted words and implied meanings cannot be handled appropriately
- NLP sometimes labels incorrectly words with multiple meanings

Conclusion

- Meta-level knowledge of the translation process from natural language into symbolic form was expressed in the same symbolic form
- Although meta-level knowledge developed was engineered, our initial results provide some confidence that it could be acquired through a process of Machine Coaching
- This, in turn, suggests that a translation process that is amenable to dynamic change in an elaboration tolerant manner is feasible

Future Work

- SWI-Prolog to be replaced by Prudens, a first-order logic language and framework
- A quantitative empirical study to evaluate the efficiency, effectiveness, and explainability of a machine-coachable translation process by working towards the following goals:
  - Demonstration of the feasibility of machine coaching for the object-level task
  - Thorough evaluation of our engineered meta-level knowledge
  - Examination on how supported sentence types can be extended
  - Extention of support to other natural languages
  - Experimentation with machine learning for the acquiring of meta-level knowledge
  - Acquisition of meta-level knowledge through verbal advice, like object-level knowledge

New Research Questions

- Certain interesting theoretical questions arise:
- Is there a meta-language rich enough to support its own verbal-coachability without the need for the involvement of experts?
- Is there a way to decide or to prove this claim?
- Or is there a need for an infinite hierarchy of meta-languages needed to support the verbal-coaching of knowledge at lower levels of the meta-language hierarchy?

Thank you!