MONA-LISA: Multimodal Ontological Neural Architecture for Linguistic Interactions and Scalable Adaptations

A Massively-Parallel Architecture for Symbolic and Subsymbolic Interactions

Hideto Tomabechi
Carnegie Mellon University
109 EDSH, Pittsburgh, PA 15213-3890, USA,
tomabech@cs.cmu.edu.
and
ATR Interpreting Telephony Research Laboratories
Seika-cho, Soraku-gun, Kyoto, JAPAN
tomabech

Abstract

This paper describes an architecture for symbolic and subsymbolic interactions during machine processing of massively-parallel cognitive activities. The model is centered around a graph-based constraint propagation network which is connected to a recurrent neural network which provides contextually sensitive predictions. The integration of symbolic massive parallelism and subsymbolic neural net PDP processing provides smooth a posteriori learning to the symbolic system and focused guided learning as well as strong constraints during recognition to the neural network. As the result, the architecture provides the ability to handle strict and structured symbolic constraints during recognition while attaining a smooth contextual prediction ability applied with the least rigidity and learning of input regularities from actual samples. The domain discussed in this paper is natural language understanding for demonstration purposes; however the architecture is expected to show equal strength in other modal channels such as visual inputs.

-49-

The strength of this architecture is that: 1) Any symbolic constraints can be represented in the constraint propagation network long as the constraints are representable using directed graphs (i.e., unification-based syntax/semantics, semantic networks, logical formulas, etc.). Therefore, both syntactic and semantic (lexicon and memory-based) contextual knowledge can be represented in a uniform framework. 2) Contextual regularities of input can be a posteriori acquired in the contextual recognition network. In the case of dialogue processing, we can provide a set of sample dialogs and the contextual recognition network learns from the actual

input. This enhances the contextual knowledge provided in the constraint propagation network since the recognition in the recurrent network is fully context sensitive whereas most of the constraints captured in the constraint propagation network are context-free. 3) Most importantly, the variety of constraints captured in the architecture is not limited to be of symbolic and subsymbolic nature. Constraints with very different nature can potentially coexist under this architecture. These constraints may vary in terms of their level of abstraction, apriority, compositionality and monotonicity, to name a few[^1]. Of course this does not exhaust the variety of constraints potentially capturable in our architecture; however, note that no previous architecture provided an uniform representation and a singular processing of any of these constraints dichotomies. In fact most of the systems simply ignored these problems. (For example, most natural language systems are purely compositional. Also very few inference systems incorporate subsymbolic information, let alone nonmonotonic ones.)

3. Graph-based Constraint Propagation Network

The main part of the MONA-LISA architecture is the Graph-based Constraint Propagation Network (GCPN). What is propagated in the GCPN are the graphs[^4] and they may point to any location in the network and may contain complex paths including convergent arcs and cycles. The expressivity of the graph-based constraint propagation scheme is significantly greater than that of so called marker-passing schemes. For example, if we want to represent the object control constraint of English in which the object of the external VP is coreferential with the subject of the embedded VP (as in [John persuaded Mary to eat sushi]), using the marker-passing scheme (such as in [Tomabechi and Levin, 1989], and [Kitano, 1989]), we will have to logically store functional applications of object control which are triggered by activations of Object control verbs. It is because in marker-passing schemes, markers can simply store bundles of feature and value pairs which are simple (i.e.,

[^1]: By different levels of abstraction I mean the differences in the nature of information from the acoustic-spectral signal level of speech (i.e. physical level) up to the abstract conceptual knowledge that may not have counterparts in the physical world.

By level of apriority I mean the nature of constraints in terms of (the degree of) the participation of experience in acquiring it, i.e., the epistemological questions about the nature of knowledge captured in our representation especially with regard to the a priori/a posteriori distinction (i.e., [Kant, 1781]). In natural language processing some constraints are a priori provided to the system while others are acquired through actual recognition of utterances.

By compositional I mean the traditional Fregean/Montagovian notion of compositionality that meaning of the whole is a function of combining the meaning of the parts. Many pragmatic constraints in natural language involve noncompositional constraint processing. Also, some constraints are configurational and others are not.

By monotonic I mean that the information is only added as time passes and never gets subtracted. Also that the information once added is never modified as time passes.

[^4]: Implementationally, they are pointers to graphs instead of graphs themselves.

-53-

(inherit-from *GIVE-ACTION*) (type :lex-head) (spelling give) (synsem (def-path (<0 loc cat head> == [[maj v] [vform bse] [aux -] [inv -] [prd -]]) (<0 loc cat marking> == unmarked) (<0 loc cat subcat 1> == <1>) (<0 loc cat subcat 2> == <2>) (<0 loc cat subcat 3> == <3>) (<0 loc cont reln> == *give-action) (<1 loc cat head maj> == n) (<1 loc cat head case> == nom) (<0 loc cont agent> == <1 loc cont para index>) (<2 loc cont restr reln> == *person) (<2 loc cat head maj> == n) (<2 loc cat head case> == acc) (<0 loc cont goal> == <2 loc cont para index>) (<2 loc cont restr reln> == *person) (❤️ loc cat head maj> == n) (❤️ loc cat head case> == acc) (<0 loc cont theme> == ❤️ loc cont para index>) (❤️ loc cont restr reln> == *entity)))

This way, instead of simply storing simple case-frame type lexical specifications in the lexical nodes, we would like to provide full graph-based lexical constraints in the lexical level nodes in the constraint propagation network. Let us provide a sample lexical rule definition for the object control verb persuaded:

(def-frame *PERSUADED (inherit-from *PERSUADE-ACTION) (type :lex-head) (spelling persuaded) (synsem (def-path (<0 loc cat head> == [[maj v] [vform inf] [aux -] [inv -] [prd -]]) (<0 loc cat marking> == unmarked) (<0 loc cat subcat 1> == <1>) (<0 loc cat subcat 2> == <2>) (<0 loc cat subcat 3> == <3>) (<1 loc cat head maj> == n) (<1 loc cat head case> == nom) (<1 loc cont restr reln> == *person) (<0 loc cont agent> == <1 loc cont para index>) (<0 loc cont persuadee> == <2 loc cont para index>) (<0 loc cont persuader> == <0 loc cont circumstance agent>) ;; obj control (<2 loc cat head maj> == n) (<2 loc cat head case> == acc) (<2 loc cont restr reln> == *person) (❤️ loc cat head maj> == v)