WHERE LEARNING AND REASONING ARE HIDDEN

PATTERN MINING

Mykola Rabchevskiy
Mar 26 ♡ 1 6

The architecture described in the previous chapter immediately raised questions: "Where is the learning module?" Where is the reasoning module?" There are no such special modules!

The entire structure of AGI serves the dual purpose of permanent learning and permanent use of accumulated knowledge for decision-making.

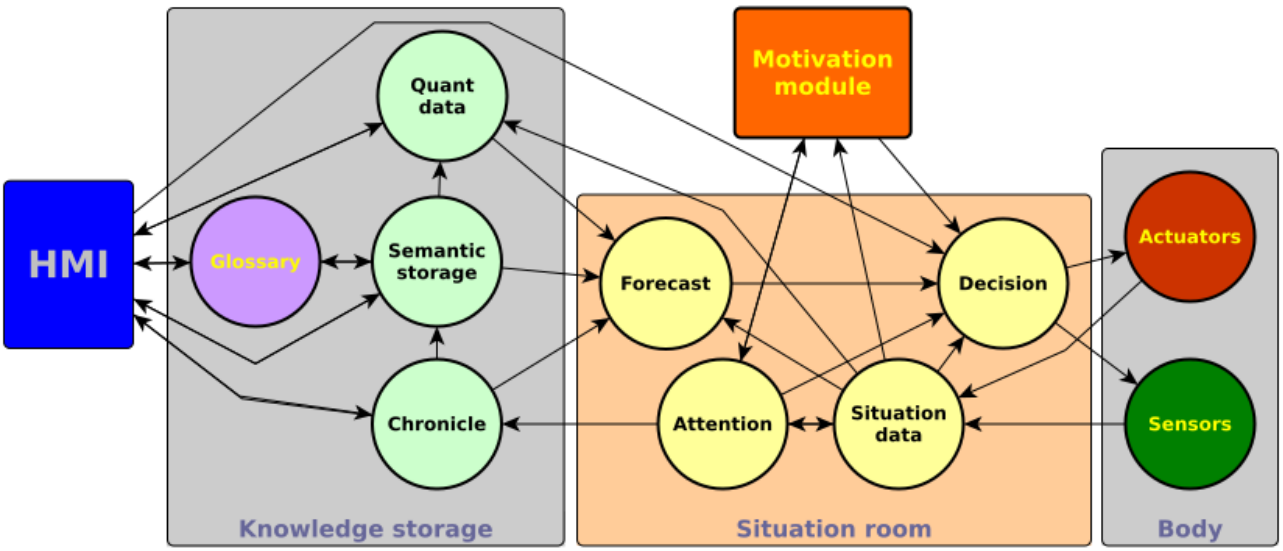
The terms ***knowledge*** and ***reasoning*** are interpreted in many ways, so it makes sense to clarify which of these variants is used in our texts. Our approach is based on the requirement that definitions are constructive, that is, on using them to construct AGI.

By ***knowledge***, we mean data about which we know how it can be used in a practical way. So far, for some data, it is not known how you can use it - it's just data. When a way of using is found, this is already knowledge.

In turn, the use of knowledge is ***reasoning***.

How can knowledge be used, that is, what kind of activity is hidden under the term

previous chapter again:



The first link in the data processing chain is assembling them into a data set that characterizes the current state (including the state of the external environment and the system itself), including sensor data fusion. Since this is a useful action on data, it is naturally a kind of reasoning (hardcoded in simple systems).

The attention module then finds the data in the state description that deserves attention. This is also an element of reasoning.

The data selected in this way add to the sequence of events in the **Chronicle**. In the Chronicle replenishment process, analysis of the sequence is performed to identify patterns, that is, repeating chains of events - this is an element of learning. The detected patterns, which are new knowledge, are stored in semantic storage and modify the Chronicle's content, increasing the share of knowledge and decreasing the share of data-not-knowledge in the chain of events.

The decision-making module's activity is reduced to the use of accumulated knowledge to make decisions - and this is in the pure form of reasoning. Since decisions are based on predictions, the forecasting module prepares data for making a decision, selecting the adequate information to the moment, which is also reasoning.

Thus, reasoning and learning are what all the ***Situation room*** modules and the ***Chronicle*** module do.

Semantic storage can also perform some sort of reasoning (search by analogy), which will be discussed in more detail in the chapter devoted to this module.

One of the most important elements of learning is the detection of repetitive chains of events (interleaved actions and their consequences) in the "biography" of the AGI system of the ***Chronicle*** module. We will refer to these repeating pieces of the overall sequence as ***patterns***.

There are many definitions for the concept of ***pattern***. Still, it is explicitly or implicitly implied that the pattern can be ***observed***; the pattern ***repeats***; the pattern is some ***combination of simpler elements***; in this case, it is a sequence of elements ordered in time, that is, a ***temporal pattern***. The ordering in time of the chain of actions and their consequences is the basis for detecting cause-and-effect relationships, which are the basis for forecasting and decision-making. Events that followed a certain action, of course, are not always the result of an action (or a sequence of actions); each individual case is nothing more than a hypothesis about the presence of a causal relationship; this aspect deserves a separate chapter.

Examples of temporal patterns are words as sequences of letters and phrases as sequences of words. Temporal patterns are also musical rhythms and melodies, body movements during walking or dancing, the sequence of technological operations, and the like.

Spatial patterns are no less common - not only ornaments, honeycombs, elements of fences, but, of course, everything that we can recognize visually, visually distinguishing some objects from others. What is less obvious is the possibility of ***reducing spatial patterns to temporal*** ones. Such a reduction constantly takes place in the activity of our brain in the process of recognizing spatial patterns: we are sequentially looking for elements of a spatial pattern, forming a sequence of operations to move from one element of a spatial pattern to another and compare them with elements of a known object. The reduction of spatial patterns to temporal patterns is a topic for a separate chapter; what is essential now is that the ability to use one-dimensional temporal patterns intelligently is useful for multidimensional cases as well.

In addition to the fact that temporal patterns are the basis for predicting the consequences of certain actions, they are also handy in that they allow you to more compactly represent the general sequence of events by replacing patterns with references to the definitions of these patterns.

This allows you to store a longer history of past events or reduce the required memory size. ***Internal definitions of specific patterns become a new logical entity (concept)***, extending the semantic component of knowledge.

It should be especially noted that there is ***no concept grounding problem in this case*** since the 'unfolded' pattern is a sequence of commands and responses from sensors, grounded by definition.

The process of discovering new patterns is combined with adding new elements to the sequence of events; discovering a new pattern is an elementary act of knowledge replenishment. If the AGI system has studied the environment utterly in which it functions, then detecting a new pattern is a signal of a possible change in the environment.

Finally, patterns can be used to exchange knowledge between AGI systems and implement a collective version of knowledge accumulation. Importing a pattern is essentially the equivalent of learning by repeating the actions of another agent.

Thus, an important aspect of implementing the learning is the pattern detection technology and how patterns are represented in the system. The ideal option is to discover a new pattern - if there is one - immediately when a new element is added to the events' sequence. This implicitly assumes that the operations for detecting new patterns are performed quickly enough.

In our case, we use a variant based on the hierarchical representation of patterns in the form of a binary tree. Each pattern is a mini-sequence of two references to logical entities, and each of them is a reference either to an elementary object or to an entity-pattern.

Obviously, the same 'unfolded' pattern of length **N** can be represented in **N-1** ways as a pair of sub-patterns. Therefore, in addition to the entity designating a pattern as a sequence of elementary events, entities are used that represent the same pattern as a sequence of two subpatterns; this category of entities is hereinafter referred to as a ***view***. At least one view is logically associated with each pattern.

This representation of patterns makes it possible to simplify the detection of new patterns after adding the next element of the sequence.

After adding a new element to the sequence, it is checked whether the last element pair does represent one of the known patterns.

To do this, a search is done for the same pair among the set of views.

If it is a known pattern, the pair is replaced with a single reference to the known pattern, shortening the sequence by one element.

If a pair does not correspond to a known pattern, an attempt is made to find a new pattern.

A pair of identical elements at the end of the sequence, if not in the set of known views, is a new

pattern.

If the elements of the last pair of elements are different, then it is checked whether this pair occurs elsewhere in the sequence.

If this is the case, this is a new pattern.

Replacing both pairs with references to the new pattern shortens the sequence by two elements.

After each replacement, it is necessary to repeat the test cycle.

The check-convolution process can take on a cascading nature when adding one element leads to creating several patterns at once and/or several substitutions of known patterns and a corresponding reduction in the length of the events' sequence.

An important aspect of such a compact representation of the sequence of events is that the last element of the sequence is essentially the current ***decision-making context*** because it represents the longest sequence of events previously encountered. This naturally resolves the question of how long the context should be used. If the last element after the possible detection of new patterns and ‘convolution’ remains an elementary object, this means that this situation is unique and did not take place before - an empty context, a totally unknown situation.

Predicting the future course of events for the current context is based on analyzing a set of patterns in which the initial subpattern coincides with the end of the virtually unfolded sequence of events. The second subpattern, in this case, is one of the possible scenarios for the subsequent course of events. Let's call the second part of the pattern ***continuation*** if the first part coincides with the tile of the 'unfolded' sequence of events. Such unfolded tail of the sequence is the unfolded last element of the packed sequence. The whole forecast is a set of possible continuations.

Finally, a few words about the most important aspects of software implementation.

To speed up the search, identical elements of the sequence are collected in linked lists; this significantly speeds up the search for new patterns since the number of references to each of the present entities is much shorter than the entire sequence.

As already noted, the sequence of events is permanently replenished, while the amount of available memory is limited. Therefore, the oldest items must be deleted if the available memory is full. For this purpose, the sequence is implemented as a circular buffer. This avoids moving data in memory while modifying the sequence. Removing elements in replacing a pair of elements with a single reference to a pattern is simulated by replacing the excluded element with an empty element.

This postpones the actual shortening of the sequence in permanent modification until the corresponding 'holes' are eliminated in deleting the oldest data in the circular buffer for inserting new elements.

SUMMATION

- *Knowledge is data that can be used to make decisions.*

- *Replenishment of knowledge with new fragments is learning.*
- *The reasoning is any manipulation of knowledge.*
- *Detection of the temporal patterns is a form of self-learning.*
- *Temporal patterns are the basis for forecasting.*
- *Temporal patterns forms innately grounded concepts.*
- *Spatial patterns can be reduced to sequential ones.*
- *Permanent mining of temporal patterns has an efficient implementation.*

← Previous Next →

Eugene Mayevski Aug 20 Liked by Mykola Rabchevskiy

" the sequence of events is permanently replenished, while the amount of available memory is limited. Therefore, the oldest items must be deleted if the available memory is full." -- events (as data or as knowledge) have different value and practicality. It makes sense to assign weight (value) to events or data and increase value if they are used and prove their usefulness. This way, not the oldest items would be deleted but the least useful ones. Of course, the value should "degrade" over time as well to let newer items stay for longer. The speed of value degradation would define the "conservatism" of the system, i.e. its favoring of older or newer data.

1 Reply

> **3 replies by Mykola Rabchevskiy and others**

Brett Martensen Mar 27 Liked by Mykola Rabchevskiy

Hi Mykola,

I love what you are doing, lots of food for thought. I just have a comment on the paragraph:

"Finally, patterns can be used to exchange knowledge between AGI systems and implement a collective version of knowledge accumulation. Importing a pattern is essentially the equivalent of learning by repeating the actions of another agent."

Since the representation of the patterns is grounded it depends on the specific configuration of the AGI's sensors and actuators. Thus the patterns can only be exchanged with another AGI that has the same sensor and actuator configuration.

♥ 1 Reply

> **1 reply by Mykola Rabchevskiy**

4 more comments…

Ready for more?

Subscribe