Knowledge transfer

Tom Rochette <tom.rochette@coreteks.org>

December 21, 2025 — 77e1b28a

0.1 Context

0.2 Learned in this study

0.3 Things to explore

- How can one be aware of information without processing it?
  - If a human has a virtual assistant (another human or an AGI), how can they share information in the most efficient manner?
- Knowledge harvesting
  - What are the proper/most efficient means of harvesting information from a person or yourself?

1 Overview

How can one transfer the models in his brain/mind in an efficient, low time-active (using as little brain cycles as possible) manner?

- Type down everything that is in one's mind in a document
- Say aloud everything that is in one's mind and record it in audio documents so that programs may be executed on the content to extract and generate models
- Write down how things are related (simple anonymous relations or highly specified relations)
- Record all written, spoken and visual content in a digital format so that programs may be executed on the content to extract and generate models
- Convert neural signals in the brain (electroencephalogram recording) into frequency functions (use BCI hardware such as OpenBCI)
- Mind uploading: Scan the whole neural substrate/brain and reproduce it in a digital format
  - Copy-and-transfer
  - Gradual replacement
- An interactive system that is able to determine stereotypes based on as few questions as possible (similar to the 20 questions game)

2 Notes on building a graph only using pairs

The premise of this section is that at a high level, the brain stores concepts and relations between them. One simple model of such structure is the pair, where two concepts are related to one another through an anonymous relation.

Here we explore how we could efficiently build a graph composed of such concepts and relations and what operations it is possible to apply on such graph.

2.1 Things of interest

- When adding new concepts, it would be interesting to record their creation date

- Easy to add/delete concepts/relations

2.2 Question

- How to separate different concepts using the same identifier?
  - Allocate a uuid?
- How to prevent duplicate entry of information (information already present)
  - Build a management system with autocompletion?
- How can you detect duplicate entries?
  - A duplicate entry should have the same identifier (not unique identifier) as another identifier, and would likely have the same relations (or one of the identifier being related to a larger set of relations than the other)
- How to reduce the amount of information that needs to be repeated
  - Use templates?
  - Use discovery in order to attempt to discover the most optimal relation structure?
- Can classes and instances be inferred simply from the data?
  - If we were to structure knowledge through a tree structure (no cycle), the most abstract concepts would be at the top (internal nodes) and the more concrete examples at the bottom (leafs). In other words, the more you are at the center of the graph (near the top of the tree), the more likely you are to be an abstract concept (class), while the further you are from the center of the graph, the more likely you are to be a concrete concept (instance)
- Can the abstractness/concreteness of a concept be assessed by its relations?
  - We expect abstract concepts to have more relations to other concepts which are concrete
  - Obviously abstractness/concreteness is relative, thus what would make the most sense is, given a concept and its related concepts, to determine whether this concept should go above (abstract) or below (concrete) its related concepts
    - One way this could be computed is to count the number of concepts the given concept is related to, and to do the same for all of its related concepts. All concepts which have more relations are considered more abstract while all concepts that have less relations are considered more concrete. All concepts which have an equal amount of concepts are considered as abstract/concrete as the given concept
    - This method may make little sense given that any concepts which can be related to many more concepts will end up being more abstract. An example of this is to have to concept for an instance of a school and concepts for each student that attended this school. Using the previously suggested algorithm would suggest to the user that this school is more abstract than all the student that attend it
- What is the best way to view this multidimensional graph?
  - Central concept with first degree relations?
    - All represented as text/list instead of a graph, where the central concept is at the top and the relations are under it
- Is there value to add a third value describing the relation between two concepts?
  - There probably is, however adding more details obviously complexifies the whole concept of the pair defining relations
  - If we accept to add a third value, then what's to prevent us from adding more values? (slippery slope argument)
- How can certainty be assessed? (some things I am 100% sure, others I can only suppose)
  - Certainty could be assessed as 100% if unspecified, otherwise a measure could be given when specifying a relation
    - A B 0.5
- How can information be extracted/explored in a structured/automated manner?
  - Templates can manually be created and assigned to concepts (not straightforward though, it would require labels (as concepts) to be attached to templates)
- Can some sort of template be constructed based on the information in the graph? (a person generally has a list of friends, a family, went to some schools, has a list of hobbies, etc.)

    - Not really as we have no idea of the type of relations that exist between concepts
    - It is possible to see certain things such that a person has link to other people (friends, family, coworkers, etc.) or some topics that are related to hobbies, sports, music or movie genre preferences, etc. however none of these relations can truly be automatically extracted
- Is the graph view the most appropriate to view and edit this type of information? Would a text visualization be as appropriate?
    - The most appropriate? At this point in time, it seems appropriate, however being the **most** appropriate, I am unsure
    - A text visualization may be as appropriate when visualizing first degree concepts, but as soon as we increase the depth of the visualization, it becomes less and less valuable/easy to understand
    - When editing first degree concepts, the main concept is “the title”, while all of its relations are “bullet points”
        * Reciprocally, if you were to click on any of the related concepts, you would be brought to that concept list of relations, which would contain the concept you came from
- Is there any value in suggesting elements related to an element that was just added?
    - In general, the answer would be no: when defining a new concept as part of a hierarchy, if it is properly placed, then it should be mutually exclusive with its siblings and be transitively related to the abstractions of the concept it was attached to
    - However a concept such as a person may relate to other people and so, by linking a person to another, it is likely that they share similar friends, parents, siblings, coworkers, etc.
- How can the most appropriate concepts be suggested while entering information about a concept? For instance, if I’m filling in information about a new person I just met, how can I get suggestions about linking this relation with “Person” and so on instead of “Anything”? If the City is transitive with the Province and then the Country, I’d rather link a person to his/her City instead of linking to all 3 concepts
    - If enough concepts are related to the newly created concept, it may be possible to attempt to do something like template matching, that is, attempt to identify other concepts which have similar relations and suggest “missing” relations
- What is the most appropriate way to store such information? Are text pairs the most appropriate way? What if I want to keep this information under version control?
    - Text pairs make sense when the size of the graph is small as it is unlikely that duplicate are present
        * However, as soon as duplicates present themselves, then we may want to be able to uniquely identify them which makes this solution less straightforward (requiring elements to be identified using some sort of unique ID)
    - Text format however is one of the most practical way to store data under version control
- If I were provided with an existing graph structure, what would be the most efficient way for me to accept/reject parts of the given graph?
    - It would most likely be required for the user to go over all of the concepts and relations to determine if he agrees or disagrees with them, however that is a huge amount of effort
    - What can be done is to go through the existing structure from the top down, and as soon as a concept is not known, then everything under it may be assumed as unknown as well
        * It is still possible to know concepts under a more abstract concept, thus the user may prefer to add some degree of uncertainty on the concepts that have not been reviewed or simply completely ignore the related concepts
- What would be some ways to minimize the amount of work/effort to build such graph?
    - Import a graph from someone else
    - Build the graph by re-using subgraph shared by others
    - Use of templates/schemas (pre-defined concepts placeholder and relations)
- What is the value in building such graph?
    - It is a list of all known concepts, which can be referenced at in the future
    - It can potentially be shared with others in order to build some sort of shared mental map (shared knowledge/concepts)
    - It can be used to describe common vocabulary
    - With higher level relational operators, it may be possible to do reasoning (induction/deduction)

and do automated consistency checking

- Is it possible to reduce the amount of time spent reviewing the graph? If so, by which means?
  - If the graph is an important tool used regularly during the day, it would make sense for its user to update it as required
  - What could be done to reduce the amount of reviewing is to use SRS (spaced repetition software) techniques/algorithms such that concepts are scheduled for review based on a given schedule. The user may specify specific schedules for given concepts in order to review them on a different schedule than general concepts
  - If it is possible, using reasoning tools the software could ask its user to fill in information regarding concepts for which this information is missing (for instance, tagging a concept as a person will automatically ask for its gender, age, hair color, eye color, etc.)
- What is the limit to the sort of relations that can be defined using only pairs?
  - These two things are related, but how? I don't know…
  - If we're willing and allowed to create many instances of the same concept, we could use a node between two concepts to specify their relation. For instance: John - Brother - Jack
    - All concepts would be related through one or many relation nodes, it would then be possible to query all concepts having this type of relation with one another
      - Adding a new type of node increases the complexity of the system and thus it would make sense to have layers of complexity
- Is it possible to consider a framework where we define at one layer the concepts, then in a second layer the anonymous relations between these concepts, and then in a third layer more complex relations and so on?

3 What is the fastest way to transfer information from the brain into text (or any app data format)?

If we are to consider not only existing, but "possible" in the future options, here is what could be done, ordered from the fastest (hypothetical) to slowest (concrete).

Quantum entanglement of every electron of the brain with a computer memory. Using this method, the computer would be able to access the memory as if it was its own, without any latency (other than accessing its own memory).

Matrix-like headjack. Similar to ethernet cables, a brain could be connected to one or many computers through a communication link. It is considered a possibility given that the spinal cord acts in a similar way (a single pathway for all signals coming from the every part of the body down the neck).

Similar to the headjack, modalities input could be recorded and sent to a computer. In this sense, the computer would have "unprocessed" signal available, but no information about what the brain decided to do with the signal.

Though identification/telepathy. Using an electroencephalogram to record brain activity, it may be possible to associate thoughts with words, effectively making it possible to convert our internal dialog into an external dialog. Hardware/software-aided telepathy would be the process of extracting the dialog in one's mind through the EEG, sending it to another person for which the signal would be translated/converted into an electromagnetic signal that this person can recognize as the original message. A potential way to get this method going would be to capture the messages as they would be about to leave the brain and go to activate the various muscles required for speech. Given a sound model of the person's voice, this information could be transmitted to the recipient where it would be processed as an audio signal, as if that person was hearing the other person if they were face to face.

Say aloud what you want to transfer to a computer and use speech recognition software. The speed of the transfer depends on the quality of the speech recognition software, namely its ability to properly identify the spoken words based on sound and semantics. Furthermore, a speech recognition software that is able to process accelerated speech will improve information transfer speed.

Type down every bit of information you want to transfer into a computer software (custom software or text editor). The speed of the transfer highly depends on the UI of the software.

4 See also

- Knowledge base

5 References