**A concept network model of a system that learns and answers questions like people**

By Zoltan Foris, [foris64@gmail.com](mailto:foris64@gmail.com)

**Motivation**

We try to build a software system with thinking capabilities equivalent to human-level intelligence. Human-level means it should be able to learn just like people (often from a single example or instruction), adapt to new tasks like people (again with little new input, or even without any new input except the task definition), produce output no worse than people, possess cognitive capabilities present in children early in development (common sense), be able to re-use anything it learned like people when facing a new task.

To launch this work, we selected natural language as single input and output. Why? We think there are a few alternatives for input and output to begin with. Practical considerations should be applied. Full human level natural language understanding and question answering is challenging enough, and useful enough, therefore having it implemented is for sure a significant step towards the goal. On the other hand, a cruicial part of the project, feeding the system with information to be learned, is by far the least resource intensive if the information is provided in natural language – especially because we aim to cover **all** domains of thinking relevant at the age of 6.

Our current goal is to implement a system with text comprehension capabilities comparable with a 6-years-old child, to be tested by the ability of the system to learn from information provided in natural language, and answer questions in natural language. This should be benchmarked against children`s performance, in all domains relevant at this age. Such system cannot work without a multi-domain common sense knowledge (intuitive theories, cognitive capabilities present early in children), therefore this is also part of the goals.

The project also aims to demonstrate the possibility to scale it to adult human level performance in natural language understanding and production.

**Summary of results**

The achievements delivered in this project so far:

1. The specification and implementation of the concept network, which is a general-purpose knowledge representation methodology.
2. A reasoning and question-answering application written in Python. This works on top of the concept network, is capable of extracting knowledge from natural language (English) text and answering questions.
3. Initial demonstration how common sense knowledge can be added to the system using natural language English sentences, and that the system is able to understand and answer questions by learning from this input.

Current work is targeted at

* expanding the common sense knowledge test with more input
* finishing the full automation of translating natural language English text into the concept network representation and vice versa.

**The concept network representation (mentalese)**

By the term *concept* we understand the following:

* On one hand, the meaning of a word or an idiomatic phrase in English is a concept
* On the other hand, one or more concepts connected with a *relation* also result a concept.

We use a set of 20 relations, some of these are:

* C : membership in a class, IsA *A bird is a kind of animal = C(bird,animal)*
* F : feature, specification, HasProperty *Joe is tall = F(Joe,tall)*
* A : action acted by actor *Joe runs = A(Joe,run)*
* IM : implication, Causes *If Joe is adult, Joe is tall* *= IM(F(Joe,adult), F(Joe,tall))*
* NOT: negation *Joe is not tall = NOT(F(Joe,tall))* or *F(Joe,NOT(tall))*. This is also an example for ambiguity – the system is designed to manage it.

The relations we use mimic the grammar of the English language. This makes it possible to develop a piece of software fairly easily which translates English into the concept network representation (which we call *mentalese*). Let us take the English sentence “Money is in the bank”. In ConceptNet, for example, the representation is something like *location(money,bank).* In our mentalese, the representation is F(money, R(in, Q(the,bank))) . The relation Q is used to relate nouns (NPs) with their articles or quantifiers. The relation R is used to relate a preposition with the following noun (NP). The meaning of the predicate *location* is carried by this portion of mentalese: *R(in* .

Features of the concept network representation (mentalese):

1. Only 20 relations; only existing English words are used, no new “words” are created.
2. Meaning is carried by the combination of (1) the original English word, (2) the position of this word in the hierarchy of nested concepts, and (3) the relations applied.
3. The resulting mentalese language is a rather mechanical copy of English, very much language-like. Therefore it is easy to translate from English to mentalese. It will be more difficult to properly translate from mentalese to English (the implementation of this part has not been started yet).
4. We can add common sense knowledge to the system using natural language English sentences. Even initially, for **knowledge children have early in their development** (intuitive physics, psychology, biology), the majority of this common sense knowledge can be fed to the system using natural English sentences. A small portion of this initial knowledge needs to be defined in mentalese directly.
5. English to mentalese translation requires rewrite rules, to be created manually.
6. Ambiguity of English is transferred to the mentalese expression. Disambiguation happens in the reasoning module which processes the mentalese expressions.
7. Multiple meanings of a word are multiple concepts. Therefore, it is not a *word* that is a concept, but *one meaning of a word.* The mentalese representation has two stages. In the first stage, word meaning is not disambiguated, and we take the word as the concept. This is the output of the mentalese translator, and the input to the reasoning module. One feature of the resoning module is to replace the word with the proper meaning.
8. The recursive nature of the concept network enables to represent any level of knowledge in the same consistent fashion. A sentence of 10 words is usually a compound concept of 3-5 levels of depth. Such sentences can be easily organized into a deeper compound concept, representing a paragraph, and this can be continued. The system will be able to deal with the concept of “a book about reasons why the US has a higher murder rate than Switzerland” with the same ease as with “Barack Obama”.
9. Concepts have a gradual truth value, for example in the range 0-4. 0 means false, 1 means might, 3 means probably, 4 means true. The sentence “Joe is probably not tall” can be represented by *NOT(F(Joe,tall))* with truth=3. This is equivalent to “Joe might be tall”, *F(Joe,tall)* with truth=1. The reasoning module performs this transformation.
10. Concepts have a number of other scalar attributes (for example, confidence level of the truth value).

Mentalese is used throughout our implementation to store all kinds of information the thinking machine needs for its operation:

* Store input (after translation from English)
* Store knowledge
* Build models
* Define reasoning rules
* Store goals, scene descriptions, event sequences etc

In fact, the above types of information are not differentiated in our system. Input is knowledge, knowledge is model, model is rule, goals, descriptions etc. A single set of processing mechanisms processes all the above.

**Reasoning and question-answering application**

We have built a reasoning and question-answering module to process the mentalese representation of the input text. Reasoning takes mentalese as input, and produces new concepts in mentalese. Question-answering takes questions in mentalese as input, and provides answers in mentalese. These are implemented (though much work is still to be done).

Short and long term memory

Inputs into the system are stored as mentalese concepts in two storages: short term memory and long term memory.

* In short term memory, each input, regardless of it beeing a duplicate of an already stored input, is being stored separately. Short term memory is regularly cleared and written into long term memory.
* In long term memory, the same concepts are not stored duplicatedly, but their properties (like occurance count, truth value, confidence of truth value etc) are merged and only one instance is stored.
  + Concepts being the same on general level but referring to different specific entities are not merged with each other, because they are not treated as being the same.

Reasoning rules

The mentalese concepts entertained by the system represent **causal models of the world**. (The “world” is the set of inputs provided to the system.) Some concepts give rise to others in causal relationships. We call this process reasoning.

Reasoning is driven by two types of mentalese concepts: (1) a small number of general rules, and (2) a large number of specific rules. Both rule types take the form of concepts, and in fact specific rules are often easy to express in English (for general rules the English version may seem odd or philosophical, as we are not used to verbalize these basic aspects of thinking).

An example of a general rule is the following. (I am using stage 1 mentalese, using the word for word meaning.) If the latest input is the concept *NOT(F(Joe,tall))*, then the system may infer the concept *F(Joe,tall)* to its short term memory, with a suitable truth value *.* This reasoning action is not hard wired, but initiated by detecting the general rule *IM(NOT(%1),%1)*. *%1* is a wildcard that matches any concept. This rule is saying that “from *NOT(something)* it follows that *something”.* The appropriate truth value of the implication is derived from the truth of the premises using a table that can be specific to the rule or shared with other rules.

So the form of a rule is always IM(A,B) where A are the premises and B is the implication. In the general rule only wildcards can be found, no specific word meaning. Special rules work in the same way but they include specific word meanings (they are often used to explain something about those meanings). They are subject to activation, see below.

Most general rules, and a portion of the special rules are added to the system directly in mentalese. This is the portion of the input that is not natural language. How many such rules might be needed? We can probably estimate this by the number of English words in use, which means that on 6-years-old level we may need a few thousand of these rules.

Activation

Reasoning uses an activation mechanism. Activation is a binary flag on the concept (or could be gradual in the future). Only activated concepts and rules take part in reasoning. This is used to avoid the exponential explosion of reasoning.

The activation mechanism enables a simple way to make the meaning of concepts sensitive to the *context.* The set of activated concepts at any given time is the current context. For example, if the system has to decide whether a spoon of salt is “much salt”, and the current set of activated concepts has a person eating a single egg by using up this amount of salt, the system will be able to correctly imply that in this context “much salt” is true.

Managing ambiguity

Ambiguity is inherent to natural language. In our model we manage ambiguity on many levels. Examples are: multiple meanings of words; multiple meanings of a natural language sentence; multiple possibilities to which an anaphor can refer; multiple meanings of a mentalese concept; multiple ways to express the same meaning in mentalese; multiple ways to translate the same mentalese to natural language. The “multiple meaning” type ambiguities are managed by allowing the evaluation of many possibilities in parallel, and selecting the best ones (this is now implemented for anaphor resolution). The “multiple ways” type ambiguities are managed currently by allowing the reasoning of several concepts with the same meaning (restrained by activation).

Exceptions, contradictions

Natural language texts are full of contradictions, which come in many forms. Two cases are currently managed in our system.

We can handle exceptions – concepts that contradict to some general knowledge. For example, the system knows that “people can talk”. Later the system learns that “people less than 1 year old usually cannot talk”. This is resolved by noting which concept is a special case of a more general concept. “people less than 1 year old” is more special than “people”, therefore reasoning with it will invalidate the reasoning with the more general concept.

The system also encounters contradictions or reinforcements not involving a generality-specificity relation. In such cases some properties of the concept (like truth value) get adjusted.

Inductive reasoning

Rules are usable to support some forms of inductive reasoning. For example, when reading about a specific cat and learning that it has a tail, the system can induce that cats in general have tails.

However more work is needed to make this really useful, and to introduce further forms of induction, for example analogical inference. This is subject to future work.

**Future work**

In order to achieve the goal of domain independent language understanding on 6-years-old level, we must provide common sense knowledge for the system. This needs to be manually selected from existing pieces of text, or even written, as there is no organized resource available for domain independent children level common sense. The effort can be significant (or even too big for a research project), however we try to write the first few dozen pages in the next 12 months for testing, and we already have the first few pages ready and working.