Concepts in Artificial intelligence

D. Gueorguiev 8/4/2020

Thought processing and Thought synthesis

Concepts and Notation

Thought Particle – a construct which does not contain start symbol and end symbol and it may be mapped to a stored thought particle. A thought particle p will be associated with a *semantic signature* $sig(p)$ which is N dimensional vector which elements are discrete sequences of real numbers with variable length. Denoted with the symbol V.

Compound thought particle – a thought particle which can be recursively partitioned in more basic thought particles. Compound particle is denoted by square brackets enclosing the contained subparticles e.g. $p = [V_1 A_1 V_2]$

Simple Thought Particle – a thought particle which cannot be subdivided further into smaller thought particles.

Connection Particle – an construct connecting two thought particles. The connection particle is a simple particle which cannot be subdivided into more elementary particles.

Thought – a construct with a starting symbol $\prec$ and an ending symbol $\succ$ which can be evaluated against a stored thought by the semantic discriminator. A thought will be associated with a direct acyclic graph (DAG) which is traceable. In this DAG there is a Hamilton path starting with start symbol $\prec$ and ending with end symbol $\succ$. Similarly to thought particle, a thought t has associated semantic signature $sig(t)$ which is N dimensional vector which elements are discrete sequences of real numbers with variable length.

Simple Thought can be presented by a path from starting symbol $\prec$ to a sequence of thought particles with an ending symbol $\succ$ appended. In other words, a *simple thought* cannot be recursively partitioned into more distinct simple thoughts without further thought transformation which would involve the outer contexts. Every thought can be described by an array of simple thoughts.

Thought/Particle Vector – condensed representation of the given particle or thought usually obtained by some sort of hashing technique resulting in multi-dimensional vector of real number sequences which uniquely represent the given particle or thought.

Thought Repository – a place for storing processed and mapped thoughts

Outer Thought Contexts – sets of processed thoughts which are not directly related to the currently processed thought.

Stored Thought – a thought which has been processed and possibly been subjected to transformation

Initial Basis – processed thoughts and particles stored already in the thought repository at the time the input stream starts flowing.

Thought Transformation – a part of the thought processing which may take place. During *thought transformation* the processed thoughts may be rearranged, coalesced or split, where each of the newly obtained thoughts is mapped against the stored thoughts accordingly.

Particle Discriminator – an adaptive learning module which processes the input stream and creates thought particles by sequencing the input stream in appropriate way.

Particle Transformer – an adaptive learning module which re-sequences and transforms thought particles into new ones which can be mapped to stored particles or to reference particles.

Thought Transformer – an adaptive learning module which re-sequences and transforms thoughts into new ones which can be mapped to stored thoughts or to reference thoughts.

Semantic Discriminator – an adaptive learning module which computes the signature of a thought and evaluates the semantic distance between two thoughts. For the purpose it builds an internal state dynamically which it uses to evaluate the thought vector(s) of the given thought(s).

Thought Harvester – harvests new thoughts after the recombination phase completes

Thought Recombiner – performs recombination event on the selected thought when triggered

Thought Ranker – assigns a dynamic score of each thought; this score is used by the thought transformer to resolve where to apply transformation.

Thought Executioner – acts on a thought marked for execution

Metric Distance Evaluator – an adaptive learning module which finds out if a given quantity e.g. a thought or a particle is small enough or if the distance between two quantities is small enough

Thought Representation

We would like to have an abstract enough and memory efficient thought representation. We do not want to embed NLP constructs into the abstract thought representation.

V_i – a thought particle is a piece of a thought represented by an N dimensional vector where $i = 1..|V|$
A thought t is a construct of attached to each other thought particles represented by a path in which the vertices are the thought particles V_i and the connections between them are given by the set of connection particles A_j where $j = 1..|A|$. Note that the connectivity between the particles models more subtle semantic nuances of the whole thought when taken in context.

Signature of thought t or particle p -

Let us consider an example:

Example 1: *I am Dimitar. My wife is Mieko. My daughters are Hanna and Emily.*

Figure 1: Possible representation
$V_1 \rightarrow$ "***I***"
$V_2 \rightarrow$ "***am***"
$V_3 \rightarrow$ "***Dimitar***"
$V_4 \rightarrow$ "***.***"
$V_5 \rightarrow$ "***My***"
$V_6 \rightarrow$ "***wife***"
$V_7 \rightarrow$ "***is***"
$V_8 \rightarrow$ "***Mieko***"
$V_9 \rightarrow$ "***.***"
$V_{10} \rightarrow$ "***My***"
$V_{11} \rightarrow$ "***daughters***"
$V_{12} \rightarrow$ "***are***"
$V_{13} \rightarrow$ "***Hanna***"
$V_{14} \rightarrow$ "***and***"
$V_{15} \rightarrow$ "***Emily***"
$V_{16} \rightarrow$ "***.***"

We will use the function $text(V)$ to denote the textual representation of the particle V. For instance $text(V_1)$ = "***I***"
Each thought particle V_i is represented by its magnitude $|V_i|$ and direction $\frac{V_i}{|V_i|}$.

Semantic Value of a Thought

A thought is represented by its *DAG (directed acyclic graph)* which in many cases degenerates to a simple path which we will denote as *thought path*. There is no intrinsic or absolute semantic value associated with a *thought path*. Instead, there is a *thought signature* which can be computed for a given thought at a given moment. The thought signature is computed with the adaptive learning module *Semantic Discriminator*. The thought signature is not static but dynamically evolves as more thoughts are stored in the repository and the thought discriminator internal state is modified in the process.
For a pair of thoughts, we can obtain a semantic distance which, similarly to the signature, is not static but dynamically changes as more thoughts are stored in the repository and the thought discriminator internal state is modified in the process.

Building Thought Path

Start with the default representation of all particles shown on Figure 1: Possible representation.
Let us assume that there are already processed and analyzed thoughts:

Figure 2: Stored thought s
"I" is a personal pronoun. Personal pronoun is a simple substitute of proper name of a person.

The built digraph for the stored thought s may look like:
$\prec V_{s_1} A_{s_1} V_{s_2} A_{s_2} V_{s_3} A_{s_3} V_{s_4} A_{s_4} V_{s_5} A_{s_5} V_{s_6} \succ$
$\prec V_{s_7} A_{s_6} V_{s_8} A_{s_7} V_{s_9} A_{s_8} V_{s_{10}} A_{s_9} V_{s_{11}} A_{s_{10}} V_{s_{12}} A_{s_{11}} V_{s_{13}} A_{s_{12}} V_{s_{14}} A_{s_{13}} V_{s_{15}} A_{s_{14}} V_{s_{16}} A_{s_{15}} V_{s_{17}} \succ$
One can infer that $V_{s_2} \simeq V_{s_8}$, $V_{s_3} \simeq V_{s_9}$, $V_{s_4} \simeq V_{s_{10}}$, $V_{s_5} \simeq V_{s_{11}}$

We are going to build and train semantic discriminator which will accept an array of thoughts t_1, t_2, … t_k. This discriminator will produce thought signatures of each of the thoughts $ssig(t_1)$, $ssig(t_2)$, …, $sig(t_k)$ and the semantic distance between every pair of thoughts t_{k_1} and t_{k_2} – $sdist(t_{k_1}, t_{k_2})$ where k_1 and k_2 are in $[1,..,k]$.
We expect for a properly trained semantic discriminator to produce small value when t_1 = $\prec V_{s_1}A_{s_1}V_{s_2}A_{s_2}V_{s_3}A_{s_3}V_{s_4}A_{s_4}V_{s_5}A_{s_5}V_{s_6} \succ$ and t_2 = $\prec V_{s_1}A_{s_1}V_{s_2}A_{s_2}V_{s_3}A_{s_3}V_{s_4}A_{s_4}V_{s_5} \succ$ as $text(V_{s_6})$ = "**.**".

Coalescing of Thought Particles
Let p_1 = $V_{s_1}A_{s_1}V_{s_2}A_{s_2}V_{s_3}$, p_2 = $V_{s_4}A_{s_3}V_{s_5}A_{s_4}V_{s_6}$
Then $p_1A_cp_2$=$V_{s_1}A_{s_1}V_{s_2}A_{s_2}V_{s_3}A_cV_{s_4}A_{s_3}V_{s_5}A_{s_4}V_{s_6}$

Coalescing of Thoughts
Let t_1 = $\prec V_{s_1}A_{s_1}V_{s_2}A_{s_2}V_{s_3}A_{s_3}V_{s_4}A_{s_4}V_{s_5}A_{s_5}V_{s_6} \succ$, t_2 = $\prec V_{s_7}A_{s_6}V_{s_8}A_{s_7}V_{s_9}A_{s_8}V_{s_{10}}A_{s_9}V_{s_{11}} \succ$ and t_3 = $\prec V_{s_{12}}A_{s_{10}}V_{s_{13}}A_{s_{11}}V_{s_{14}} \succ$
Simple concatenation:
Then t_1t_2=$\prec V_{s_1}A_{s_1}V_{s_2}A_{s_2}V_{s_3}A_{s_3}V_{s_4}A_{s_4}V_{s_5}A_{s_5}V_{s_6} \succ \prec V_{s_7}A_{s_6}V_{s_8}A_{s_7}V_{s_9}A_{s_8}V_{s_{10}}A_{s_9}V_{s_{11}} \succ$
Splice operation:
$t_1t_2 \vdash_{t_1} t_3$= $\prec V_{s_1}A_{s_1}V_{s_2}A_{s_2}V_{s_3}A_{s_3}V_{s_4}A_{s_4}V_{s_5}A_{s_5}V_{s_6} \succ \prec V_{s_{12}}A_{s_{10}}V_{s_{13}}A_{s_{11}}V_{s_{14}} \succ \prec V_{s_7}A_{s_6}V_{s_8}A_{s_7}V_{s_9}A_{s_8}V_{s_{10}}A_{s_9}V_{s_{11}} \succ$
Split operation:
$t_1t_2 \backslash t_1$=$\prec V_{s_7}A_{s_6}V_{s_8}A_{s_7}V_{s_9}A_{s_8}V_{s_{10}}A_{s_9}V_{s_{11}} \succ$
$t_1t_2 \vdash_{t_1} t_3 \backslash t_2$= $\prec V_{s_1}A_{s_1}V_{s_2}A_{s_2}V_{s_3}A_{s_3}V_{s_4}A_{s_4}V_{s_5}A_{s_5}V_{s_6} \succ \prec V_{s_{12}}A_{s_{10}}V_{s_{13}}A_{s_{11}}V_{s_{14}} \succ$

If $sdist(t_1t_2, t_c) < \varepsilon(t)$ then we are going to refer to t_1t_2 as t_c.

Comparing Thoughts
Let t_1 be a thought represented by path P_1 and let t_2 be a thought represented by path P_2.
Let t_1 has signature S_1 and t_2 has signature S_2. Then t_1 and t_2 will be equivalent if S_1 and S_2 are identical or if $sdist(S_1, S_2)$ is small enough.

Calculating the semantic signature of a thought t :
$ssig(t) = \prec (\omega) + V_1(\omega)\delta(\omega - \Delta\omega_1) + A_1(\omega)\delta(\omega - \Delta\omega_2) + V_2(\omega)\delta(\omega - \Delta\omega_3) + \cdots + V_k(\omega)\delta(\omega - \Delta\omega_{2k-1}) + \succ (\omega)$ where $\omega \in [0, N_t]$.
Here $V_l(\omega)$ represents the particle vector for the thought particle V_l, $l = 1,..,k$.

<u>Example</u> 2: *John is the father of Sam. Julie is the mother of Sam. If a person is your father and another person is your mother then you are their son. → Sam is son of John and Julie.*
Thought path of the intermediate representation
$\prec V_1A_1V_2A_2V_3 \succ \prec V_4A_3V_5A_4V_6 \succ \prec V_7A_5V_8A_6[V_9A_7V_{10}]A_8V_{11}A_9[V_{12}A_{10}V_{13}]A_{11}V_{14}A_{12}V_{15} \succ$
$text(V_1) = "John"$
$text(A_1) = "is"$
$text(V_2) = "the\ father"$
$text(A_2) = "of"$

$text(V_3) = "Sam"$
$text(V_4) = "Julie"$
$text(A_3) = "is"$
$text(V_5) = "the\ mother"$
$text(A_4) = "of"$
$text(V_6) = "Sam"$
$text(V_7) = "If"$
$text(A_5) = ""$
$text(V_8) = "a\ person"$
$text(A_6) = "is"$
$text(V_9 A_7 V_{10}) = "your\ father"$
$text(A_8) = "and"$
$text(V_{11}) = "another\ person"$
$text(A_9) = "is"$
$text(V_{12} A_{10} V_{13}) = "your\ mother"$
$text(A_{11}) = "then"$
$text(V_{14}) = "you"$
$text(A_{12}) = "are"$
$text(V_{15}) = "their\ son"$

By *the Laws of Attraction* we have
$\prec V_1 A_1 [V_2 A_2 V_3] \succ$

If $\|key(p_1, prop) - key(p_2, prop)\| < \varepsilon(key, prop)$ we say that the keys of particles p_1 and p_2 match over property $prop$. In such case the two particles p1 and p2 will attract each other with *focal point of attraction* the property $prop$.
If $\|key(p_1, prop) - key(p_2, prop)\| > \Delta(key, prop)$ we say that the keys of particles p_1 and p_2 are mismatched over property $prop$. In such case the two particles p_1 and p_2 will repulse each other with *focal point of repulsion* the property $prop$.
Particle property – a region on one of the dimensions of the particle signature which exhibits specific non-trivial shape.
Particle locking over a property – a region in the signature exists which allows locking by another particle which exhibits interlocking pattern on a region corresponding to the same property.
Attraction force between two particles – proportional to the weight of the interlocking regions for the two particles.
Repulsion force between two particles – proportional to the weight of the mismatched regions for two particles.

Recombination of particles, laws of attraction and repulsion, inference

The laws of attraction and repulsion between particles have to be formulated in such a way that will make semantic inference possible and the inference will be more accurate with increasing the learning experience.

Let us go back to our Example 1: *I am Dimitar. My wife is Mieko. My daughters are Hanna and Emily.* Here is our initial DAG representation of this thought:
$\prec V_1 A_1 V_2 \succ\prec [V_3 A_2 V_4] A_3 V_5 \succ\prec [V_6 A_4 V_7] A_5 [V_8 A_6 V_9] \succ$

$text(V_1)$=*"I"*

$text(A_1)$=*"am"*
$text(V_2)$=*"Dimitar"*
$text(V_3)$=*"My"*
$text(A_2)$=*""*
$text(V_4)$=*"wife"*
$text(A_3)$=*"is"*
$text(V_5)$=*"Mieko"*

Example 3: *George has two bottles of wine. Shirley has two bottles of wine. How many bottles of wine together do George and Shirley have? -> George and Shirley have two bottles of wine and two bottles of wine together.*
<V1A1[V2A2V3]><V4A3[V5A4V6]><V7A5V8A6V9A7V10>

Updating Semantic Discriminator through training

Naïve approach
Let us have repository with stored thoughts t_1, t_2, t_3, … , t_k
//TODO: finish this

D. Gueorguiev 8-4-2020