Zihinde anlam bilişimle oluşabilir mi?

Cem Bozşahin
Bilişsel Bilimler
Enformatik Enstitüsü
ODTÜ
bozsahin@metu.edu.tr

ODTÜ DAS

14.2.2015
Can Computation Give Rise to Meaning?

Cem Bozşahin
Cognitive Science Department
ODTÜ
bozsahin@metu.edu.tr
Turing (1950): Check it out.
   Semantics as verbal behavior
   Computation is purely formal (syntactic).
   We need the right stuff (brain) to cause semantics.
Rapaport (1986): Yes.
   Syntactic semantics (tripartite compositional semantics)
   \[2x + 4 = 5\] has syntactical semantics and “physical” semantics
   (tripartite relation)
   This is not a wetware/hardware problem.
   Program qua algorithm does not understand, but the running
   process does.
Me: Even so, there is a limit to the kinds of meanings
computation can provide.
Turing’s imitation game

Searle-in-the-box: Chinese Room

The original thought experiment is Rogers (1959).
CR is an ill-thought experiment: No grammar can be turned into a look-up table of forms and fit into a finite-size room. Bozşahin (2006, 2012)

Rey (1986); Rapaport (2006)

If humans only exchanged forms like CR, they could not learn meanings either.

They appear to triangulate forms with some verbal or bodily behavior and interaction.
Did Helen Keller learn to cause semantics? Rapaport (2006)

How Helen Keller used syntactic semantics to escape from a Chinese Room

Two channels of information:
- forms
- behavior, action, and observation (of the world, internal and external)
Helen Keller, 1880–1968
- Children with autism manifest soliloquy.
- Blind children learn the difference between look and watch: Landau and Gleitman (1985): *you can touch the table but don’t look at it!*
- Deaf children acquire sign language (not gestures) **If** they are exposed to data in the critical period, just like other children.
Grammar, cognition and computation: bird’s eye view
Searle’s unrealistic Chinese Room
Helen Keller’s Room
Can Computation Give Rise to Meaning?

Logician’s view of: every monk owns a ferrari

\[(\forall x)(\text{monk}(x) \rightarrow (\exists y)(\text{ferrari}(y) \land \text{owns}(x, y)))\]

\[(\exists y)(\text{ferrari}(y) \land (\forall x)(\text{monk}(x) \rightarrow \text{owns}(x, y)))\]
Linguist’s view

Heads of substantive phrases have lexical content:

```
S
  NP       VP
  every    owns    NP
        monk   a   ferrari
                 own'  exist'  ferrari'
                           every'  monk'
```

What are primes about?
Can Computation Give Rise to Meaning?

Computational Linguist’s view

All head dependencies are efficiently computable

Where do primes come from? What causes semantics?

How does a word come to be about a prime?
Can Computation Give Rise to Meaning?

Projecting structure

S

: (∀x₁)(monk′x₁ → (∃x₅)ferrari′x₅ ∧ owns′x₅x₁)

NP

: λQ.(∀x₁)(monk′x₁ → Qx₁)

every

: λP λQ.(∀x₁)(Px₁ → Qx₁)

monk

: λx₂.monk′x₂

: λx₃ λx₄.own′x₃x₄

owns

: λQ.(∃x₅)(ferrari′x₅ ∧ Qx₅)

NP

ferrari

a

: λP λQ.(∃x₅)(Px₅ ∧ Qx₅)

: λx₆.ferrari′x₆
Meaning without lexical content of words

Lexical content cannot be predicted from grammar.
Meaning without grammar

<table>
<thead>
<tr>
<th>every</th>
<th>monk</th>
<th>owns</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\forall x_1 (P_{x_1} \rightarrow Q_{x_1})$</td>
<td>$\lambda x_2.\text{monk}'x_2$</td>
<td>$\lambda x_3 \lambda x_4.\text{own}'x_3 x_4$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>a</th>
<th>ferrari</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\exists x_5 (P_{x_5} \land Q_{x_5})$</td>
<td>$\lambda x_6.\text{ferrari}'x_6$</td>
</tr>
</tbody>
</table>

Structure of meaning cannot be predicted without a grammar.
Meaning from word-grammar

\[
\begin{align*}
S & \rightarrow \text{NP } \text{VP} & S' &= \text{NP}'(\text{VP}') \\
\text{VP} & \rightarrow \text{V } \text{NP} & \text{VP}' &= \text{NP}'(\text{V}') \\
\text{NP} & \rightarrow \text{Det } \text{N} & \text{NP}' &= \text{Det}'(\text{NP}') \\
\text{Det} & \rightarrow \text{every} & \text{Det}' &= \lambda P \lambda Q.(\forall x)(Px \rightarrow Qx) \\
\text{Det} & \rightarrow \text{a} & \text{Det}' &= \lambda P \lambda Q.(\exists x)(Px \land Qx) \\
\text{N} & \rightarrow \text{monk} & \text{N}' &= \lambda x.\text{monk}'x \\
\text{N} & \rightarrow \text{ferrari} & \text{N}' &= \lambda x.\text{ferrari}'x \\
\text{V} & \rightarrow \text{owns} & \text{V}' &= \lambda x \lambda y.\text{owns}'xy
\end{align*}
\]

Top part projects, and bottom part initiates meaning (hence the dichotomy).

Can we predict structure and lexical content together?

A causal mechanism for expressible/expressed meanings
Can Computation Give Rise to Meaning?

Reducing a grammar to its lexicon without loss of structure

Every right-hand side has one symbol; such rules are functions looking from constituent’s perspective.

\[
\begin{align*}
S & \rightarrow \ NP \ VP & S' = NP'(VP') \\
VP & \rightarrow \ V \ NP & VP' = NP'(V') \\
NP & \rightarrow \ Det \ N & NP' = Det'(NP')
\end{align*}
\]

\[
\begin{align*}
NP = S/ (S\setminus NP) & \quad V = (S\setminus NP)/ NP & \quad NP = (S\setminus NP)/(S\setminus NP)/NP) \\
Det = (S/(S\setminus NP))/N & \quad Det = ((S\setminus NP)/(S\setminus NP)/NP))/N \\
N = (S/(S\setminus NP)/(S\setminus NP))/N) & \quad N = ((S\setminus NP)/(S\setminus NP)/NP)/(S\setminus NP)/(S\setminus NP)/NP)/N)
\end{align*}
\]

Slashed cats: structure-equivalent combinatory categories (eqv. under substitution)
We’ve got N, V, Det without a need for NP, VP or S rule

N, V, Det are the only lexical categories in the grammar!

\[
\begin{align*}
S & \rightarrow \ NP \ VP \\
VP & \rightarrow \ V \ NP \\
NP & \rightarrow \ Det \ N \\
\text{Det} & \rightarrow \ \text{every} \\
\text{Det} & \rightarrow \ \text{a} \\
N & \rightarrow \ \text{monk} \\
N & \rightarrow \ \text{ferrari} \\
V & \rightarrow \ \text{owns}
\end{align*}
\]
Two grammars capture the same structures and meanings:

\[
\begin{align*}
S & \rightarrow \ NP \ VP & S' = NP'(VP') \\
VP & \rightarrow \ V \ NP & VP' = NP'(V') \\
NP & \rightarrow \ Det \ N & NP' = Det'(NP') \\
Det & \rightarrow \ every & Det' = \lambda P \lambda Q \cdot (\forall x)(P x \rightarrow Q x) \\
Det & \rightarrow \ a & Det' = \lambda P \lambda Q \cdot (\exists x)(P x \land Q x) \\
N & \rightarrow \ monk & N' = \lambda x \cdot monk' x \\
N & \rightarrow \ ferrari & N' = \lambda x \cdot ferrari' x \\
V & \rightarrow \ owns & V' = \lambda x \lambda y \cdot owns' xy \\
\end{align*}
\]

\[
\begin{align*}
evvery & := \frac{S}{(S \setminus NP)} \setminus N & : \lambda P \lambda Q \cdot (\forall x)(P x \rightarrow Q x) \\
\text{a} & := \frac{(S \setminus NP) \setminus ((S \setminus NP) \setminus NP)}{N} & : \lambda P \lambda Q \cdot (\exists x)(P x \land Q x) \\
\text{monk} & := N & : \lambda x \cdot monk' x \\
\text{ferrari} & := N & : \lambda x \cdot ferrari' x \\
\text{owns} & := \frac{S \setminus NP}{NP} & : \lambda x \lambda y \cdot owns' xy \\
\end{align*}
\]

Difference: in the red corner, form-meaning relation only through words.
Can Computation Give Rise to Meaning?

owns := (S\NP)/NP : λxλy.owns'xy
Real semantics arising from computation is probably not the proxy objects like *monk’, *ferrari’, *every’,

But the *process* of their construction.
Learning veggies are veggies, eating is eating, plural is plural

Eat veggies.

possible hypotheses:

\[
\begin{align*}
\text{eat} & := S/\text{NP}: \text{eat}' & \text{veggies} & := \text{NP}: \text{veg}' \\
\text{eat} & := S/\text{NP}: \text{eat}' & \text{veggie} & := \text{NP}: \text{veg}' & -s & := \text{NP}\setminus \text{NP}: \text{plu}' \\
\text{eat} & := \text{NP}: \text{eat}' & \text{veggies} & := S\setminus \text{NP}: \lambda x. \text{veg}'x \\
\text{eat} & := \text{NP}: \text{veg}' & \text{veggies} & := S\setminus \text{NP}: \lambda x. \text{eat}'x \\
\text{eat} & := S/\text{NP}: \text{eat}' & \text{veggie} & := \text{NP}/\text{NP}: \text{plu}' & -s & := \text{NP}: \text{veg}' \\
\end{align*}
\]

impossible hypotheses:

\[
\begin{align*}
* \text{eat} & := \text{NP}: \text{eat}' & \text{veggies} & := S/\text{NP}: \text{veg}' \\
* \text{eat} & := S\setminus \text{NP}: \text{eat}' & \text{veggies} & := \text{NP}: \text{veg}' \\
* \text{eat} & := S\setminus \text{NP}: \text{eat}' & \text{veggie} & := \text{NP}: \text{veg}' & -s & := \text{NP}\setminus \text{NP}: \text{plu}' \\
\end{align*}
\]
<table>
<thead>
<tr>
<th>Experience 1 (Eat veggies)</th>
<th>Experience 2 (No veggies; with chocolate)</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>eat := S/ NP: eat'</code></td>
<td><code>no := S/ NP: no'</code></td>
</tr>
<tr>
<td></td>
<td><code>veggies := S\ NP: veg'</code></td>
</tr>
<tr>
<td></td>
<td><code>veggie := NP : veg'</code></td>
</tr>
<tr>
<td></td>
<td><code>-s := NP\ NP: plu'</code></td>
</tr>
<tr>
<td></td>
<td><code>NP : veg'</code></td>
</tr>
<tr>
<td></td>
<td><code>: eat'</code></td>
</tr>
<tr>
<td><code>NP : eat'</code></td>
<td><code>: veg'</code></td>
</tr>
<tr>
<td></td>
<td><code>: plu' veg'</code></td>
</tr>
<tr>
<td><code>NP : veg'</code></td>
<td><code>: plu' eat'</code></td>
</tr>
<tr>
<td></td>
<td><code>NP : veg'</code></td>
</tr>
<tr>
<td></td>
<td><code>: eat'</code></td>
</tr>
<tr>
<td></td>
<td><code>: plu' veg'</code></td>
</tr>
<tr>
<td></td>
<td><code>: plu' eat'</code></td>
</tr>
<tr>
<td><code>NP : veg'</code></td>
<td><code>NP : veg'</code></td>
</tr>
<tr>
<td><code>: eat'</code></td>
<td><code>: veg'</code></td>
</tr>
<tr>
<td></td>
<td><code>: plu' veg'</code></td>
</tr>
<tr>
<td><code>NP : veg'</code></td>
<td><code>: plu' eat'</code></td>
</tr>
<tr>
<td><code>NP : veg'</code></td>
<td><code>NP : veg'</code></td>
</tr>
<tr>
<td><code>: eat'</code></td>
<td><code>NP : veg'</code></td>
</tr>
<tr>
<td></td>
<td><code>: plu' veg'</code></td>
</tr>
<tr>
<td></td>
<td><code>: plu' eat'</code></td>
</tr>
<tr>
<td><code>NP : veg'</code></td>
<td><code>NP : veg'</code></td>
</tr>
<tr>
<td><code>: eat'</code></td>
<td><code>NP : veg'</code></td>
</tr>
<tr>
<td></td>
<td><code>: plu' veg'</code></td>
</tr>
<tr>
<td><code>NP : veg'</code></td>
<td><code>NP : veg'</code></td>
</tr>
<tr>
<td><code>: eat'</code></td>
<td><code>NP : veg'</code></td>
</tr>
<tr>
<td></td>
<td><code>: plu' veg'</code></td>
</tr>
</tbody>
</table>

Even in this circumscribed world of two experiences only, the child is exponentially less likely to believe that veggies could mean negation, eating, plural or chocolate, rather than veggies.
Veggies gone.

veggies := $S/\text{NP}:\text{veg}'$
  \cdot \text{gone}'
  \cdot \text{eat}'
  \cdot \text{no}'
  \cdot \text{plu}'\text{veg}'$
\text{NP}
  \cdot \text{veg}'
  \cdot \text{gone}'
  \cdot \text{no}'
  \cdot \text{plu}'
\cdot \text{gone}'$
\cdot \text{eat}'
\cdot \text{no}'
\cdot \text{plu}'$
\text{gone}'$
\cdot \text{no}'$
\cdot \text{plu}'$
\cdot \text{gone}'$
\cdot \text{no}'$
\cdot \text{plu}'$
\cdot \text{gone}'$
\cdot \text{no}'$
\cdot \text{plu}'$
\{veggies, veggie\} := \{  
S\ NP:veg'@2_{55}, S\ NP:eat'@2_{55}, S\ NP:no'@1_{55}, S\ NP:plu' eat'@1_{55},  
S\ NP:choc'@1_{55}, S\ NP:plu' veg'@2_{55}, S\ NP:plu' choc'@1_{55}, S/ NP:plu' veg'@1_{55},  
S\ NP:plu' no'@1_{55}, S\ NP:plu' choc'@1_{55}, S/ NP:plu' no'@1_{55}, S/ NP:plu' gone'@1_{55},  
NP:veg'@9_{55}, NP:gone'@4_{55}, NP:choc'@3_{55},  
NP:plu' eat'@1_{55}, NP:plu' gone'@1_{55}, NP:plu' choc'@1_{55},  
NP:plu' no'@1_{55}, NP:no'@4_{55},  
NP:plu' @3_{55}, NP:plu' @1_{55}, NP:plu' @1_{55},  
NP/ NP:veg'@3_{55}, NP/ NP:plu'@3_{55}, NP/ NP:plu'@3_{55},  
\}
Other experiences with approximate but probable meanings

- Planning
- Music
- Vision
- Art

- All high-level cognitive processes are massively serial.
- All low-level processes are massively parallel.
- Need for symbols seems to be the key (not in Beckett’s sense) for the bottleneck (Deacon 1997, 2012).
- Unexpected contribution of grammars in all these domains
- All we need to engender meaning of this sort is a mechanism to execute the grammatical process.
Humans are doing computations too, when they learn grammar and words.

Searle is a pessimist, and Turing an optimist, about artificial systems doing the same thing.

Cognitive science, esp. computational linguistics, show how the process can be conceived computationally for humans, and for other things with interpretable hardware.

That’s their “right stuff.”
A grammar-parser without delivery of meaning is a non-starter.

We can accuse current artificial systems of not doing anything interesting by way of semantics.

That doesn’t mean they are incapable.
mechanistic explanations

- Dualism of Descartes, and classical mechanics of Newton saw an insurmountable divide between deterministic mechanisms and free will/nondeterminism. Gorham (2011)
- Descartes believed they must be physically mediated somehow.
- Quantum mechanics changed all that: body is just as nondeterministic.
- cf. Chomsky (2000), who says notion of body is unclear (so we should study the mental?). Lycan (2003)
- Nagel (1986): Two aspects, rather than two insurmountable worlds.
- Maybe the real question is: what are the mechanisms involved?
Sad but true

- There are uncountably many meanings out there.
- In a grammar, we can express countably infinitely many.
- Some meanings cannot be expressed.
- The kind of meanings that can be expressed cause the same problems for the owners with the right hardware:
  - ambiguity in perception and use
  - indeterminacy and likelihood
  - resource boundedness
Concluding remarks

- If we worry about the complexity of the problem, and its mechanism,
- computation as we know today can give rise to PAC meanings*

  - Only they can be given a causal history of their construal with reasonable resources.
  - Valiant (1984): “Inherent algorithmic complexity appears to set serious limits to the range of concepts that can be learned.”
  - Perhaps only computation can give rise to such meanings.

*Probably Approximately Correct. Valiant 1984: “we regard learning as the phenomenon of knowledge acquisition in the absence of explicit programming.” The selected hypothesis has high probability for low generalization error.
There is a Kantian feel to it: hypotheses may be close to reality, but as a proxy.

If there is more to meaning than that, look somewhere else.

“In seeing ourselves from outside we find it difficult to take our lives seriously. This loss of conviction, and the attempt to regain it, is the problem of the meaning of life.”

Nagel 1986:214

We can do science of mind without scientism.
Transfinite representations can be talked about (e.g. $\pi$), but cannot be pinned down (not even the PAC-way).

Can we look into the brain and see the meaning?

- Probably not
- But we might be able to construct a personal history for a meaning associated with a form.
- Such histories not only contain winning solutions, but also ill-fated attempts, non-attempts, and near-solutions.
- A theory of other minds tells us like minds can do similar things.

It is a process.
Box 1. Knowing what others see

A dominant and a subordinate chimpanzee compete for food, with only the subordinate having information about the location of the second piece of food [23] (Fig. I).

Method: Subordinate (S) and Dominant (D) chimpanzee in rooms on opposite sides of a middle room, each looking under cracked guillotine door to see food (F) and each other. S and D released into middle room, with S given a brief head start.

Main experiment: S and D see one F in open and only S sees other F on her side of barrier.

Result: S goes for F on her side of barrier more than F in open.

Interpretation: S knows what D can and cannot see.

Alternative hypotheses and control conditions

1. S is reacting to D’s behavior. But S has a head start and so is forced to choose before D’s door is opened.
2. D looks under door at F in open and intimidate S. But sometimes D’s door is down when S makes her choice and so intimidation is not possible.
3. S prefers F next to a barrier. But when S is given a choice of F in open or F on her side of a barrier in non-competitive situations, no preference.
4. S thinks barrier impedes D in getting F. But when F is on S’s side of a transparent barrier, S stays away from it.
Box 2. Knowing what others know

A dominant and a subordinate chimpanzee compete for food, with either one or both of them (in different experimental conditions) observing the hiding process [24] (Fig. I).

Method: same as in Box 1 except that there are two barriers and one F (on S’s side of one barrier).

Main experiment: (a) Control condition: S and D watch hiding process; (b) Experimental condition: only S watches hiding process

Result: S goes for F more in Experimental condition than Control condition.

Interpretation: S knows what D has and has not seen in immediate past.

Alternative hypotheses and control conditions

1. S prefers F next to a barrier when competing. But in both conditions F is situated identically – next to a barrier.
2. When D watches hiding she puts evil eye on F being hidden and S stays intimidated. But (a) in a separate experiment the D who has witnessed the hiding process is switched (or not, in a control condition) for another D who has not witnessed the hiding process, and S goes for F more when D is switched than when not; and (b) in a separate experiment S and D watch F being hidden but only S watches it being moved to a new location – and S does not then avoid it even though D has looked at it previously.

Fig. I. Experimental set-up in the second set of food-competition experiments. Reproduced with permission from Ref. [4].
Can Computation Give Rise to Meaning?


