EMERGENCE OF INTELLIGENT MACHINES: CHALLENGES AND OPPORTUNITIES

CS6700:
The Emergence of Non-Human Intelligence

Prof. Carla Gomes
Prof. Bart Selman
Cornell University
AI focus: **Human** intelligence because that’s the intelligence we know…

Cognition: Perception, learning, reasoning, planning, and knowledge.

Deep learning is changing what we thought we could do, at least in perception and learning (with enough data).
Separate development --- “non-human”: Reasoning and planning. Similar qualitative and quantitative advances but “under the radar.”

Part of the world of software verification, program synthesis, and automating science and mathematical discovery.

Developments proceed without attempts to mimic human intelligence or even human intelligence capabilities.

Truly machine-focused (digital): e.g., “verify this software procedure” or “synthesize procedure” --- can use billions of inference steps --- or “synthesize an optimal plan with 1,000 steps.” (Near-optimal: 10,000+ steps.)
Example

Consider a sequence of 1s and -1s, e.g.:

-1, 1, 1, -1, 1, 1, -1, 1, -1 ...

1 2 3 4 5 6 7 8 9 ...

2 4 6 8 ...

3 6 9 ...

and look at the sum of sequences and subsequences:

\[
\begin{align*}
-1 + 1 &= 0 \\
-1 + 1 + 1 &= 1 \\
-1 + 1 + 1 + -1 &= 0 \\
-1 + 1 + 1 + -1 + 1 &= 1 \\
-1 + 1 + 1 + -1 + 1 + 1 &= 2 \\
-1 + 1 + 1 + -1 + 1 + 1 + -1 &= 1 \\
-1 + 1 + 1 + -1 + 1 + 1 + -1 + 1 &= 2 \\
-1 + 1 + 1 + -1 + 1 + 1 + -1 + 1 + -1 &= 1 \\
& \text{etc.}
\end{align*}
\]

and “skip by 1”

\[
\begin{align*}
1 + -1 &= 0 \\
1 + -1 + 1 &= 1 \\
1 + -1 + 1 + 1 &= 2 \\
& \text{etc.}
\end{align*}
\]

and “skip by 2”

\[
\begin{align*}
1 + 1 &= 2 \\
1 + 1 + -1 &= 1 \\
& \text{etc.}
\end{align*}
\]

We now know (2015): there exists a sequence of 1160 +1s and -1s such that sums of all subsequences never < -2 or > +2.
1160 elements
all sub-sums stay between -2 and +2

40 x 29 pattern
So, we now know (2015): there exists a sequence of 1160 +1s and -1s such that sums of all subsequences never < -2 or > +2.

Result was obtained with a general reasoning program (a Boolean Satisfiability or SAT solver). Surprisingly, the approach far outperformed specialized search methods written for the problem, including ones based on other known types of sequences. (A PolyMath project started in January 2010.)
Aside: A Taste of Problem Size

Consider a real world Boolean Satisfiability (SAT) problem, from software & hardware verification.

The instance bmc-ibm-6.cnf, IBM LSU 1997:

```
-1 7 0
-1 6 0
-1 5 0
-1 -4 0
-1 3 0
-1 2 0
-1 -8 0
-9 15 0
-9 14 0
-9 13 0
-9 -12 0
-9 11 0
-9 10 0
-9 -16 0
-17 23 0
-17 22 0
```

“1” for variable x_1, “2” for x_2, etc.

x_1, x_2, x_3, … our Boolean variables (set to True or False)

((not x_1) or x_7)
((not x_1) or x_6)

etc.

Question: Can we satisfy all statements?

Set x_1 to False ??

SAT problem lies at the core of computer science

Prototypical NP-complete problem (from P vs. NP)
I.e., \((x_{177} \text{ or } x_{169} \text{ or } x_{161} \text{ or } x_{153} \ldots \text{ or } x_{33} \text{ or } x_{25} \text{ or } x_{17} \text{ or } x_{9} \text{ or } x_{1} \text{ or } \neg x_{185})\)

clauses / constraints are getting more interesting…

Note \(x_1\) …
4000 pages later:

\[
\begin{align*}
10236 & -10050 0 \\
10236 & -10051 0 \\
10236 & -10235 0 \\
10008 & 10009 10010 10011 10012 10013 10014 \\
10015 & 10016 10017 10018 10019 10020 10021 \\
10022 & 10023 10024 10025 10026 10027 10028 \\
10029 & 10030 10031 10032 10033 10034 10035 \\
10036 & 10037 10086 10087 10088 10089 10090 \\
10091 & 10092 10093 10094 10095 10096 10097 \\
10098 & 10099 10100 10101 10102 10103 10104 \\
10105 & 10106 10107 10108 -55 -54 53 -52 -51 50 \\
10047 & 10048 10049 10050 10051 10235 -10236 0 \\
10237 & -10008 0 \\
10237 & -10009 0 \\
10237 & -10010 0 \\
\end{align*}
\]
Finally, 15,000 pages later:

Search space of truth assignments:

$$2^{50000} \approx 3.160699437 \cdot 10^{15051}$$

Current reasoning engines can solve this instance in a few seconds! (no satisfying assignment exists + proof)
Back to sequences of +1/-1s

Encoding has variables for the sequence $X_1, X_2, \ldots, X_N$
(we interpret True for +1 and False for -1)

but also e.g.

Proposition: “sum_of_first_2_terms_of_step_by_2_subseq_=2”
(for any given setting of $X_1 \ldots X_N$ this is either True or False)

and statements of the form:

IF ((sum_of_first_2_terms_of_step_by_2_subseq_=2 == True)
AND (X_8 == False))
THEN
(sum_of_first_3_terms_of_step_by_2_subseq_=1 == True)

Encoding: 37,418 variables and 161,460 clauses / constraints.

Sequence found in about 1 hour (MacBook Air).
Perhaps SAT solver was “lucky” in finding the sequence?
But, remarkably, each sequence of 1161 or longer leads to a +3 (or -3) somewhere. (Erdos discrepancy conjecture)

Encoding: 37,462 variables and 161,644 clauses / constraints.

Proof of non-existence of discrepancy 2 sequence found in about 10 hour (MacBook Air).

Proof: 13 gigabytes and independently verified (50 line proof checking program). Proof is around a billion small inference steps.

Machine understands and can verify result easily (milliseconds); Humans: probably never. Still, we can be certain of the result because of the verifier.
Observations

1) Result different from earlier “computer math” results, such as the proof of the 4 color theorem, because here we don’t need to trust the theorem prover. Final proof (“certificate”) can be checked easily by anyone.

2) It’s not a brute force search. Earlier SAT solvers cannot find the proof. Specialized programs cannot find the proof.

Brute force proof is of order $2^{1161} = 3.13 \times 10^{349}$. Current solver finds complete proof with “only” around $1.2 \times 10^{10}$ steps. Clever learning and reasoning enables a factor $10^{339}$ reduction in proof size.

3) In part inspired by discrepancy 2 result, Terence Tao proved just a few months ago the general Erdos conjecture (for any discrepancy). Deep and subtle math.

4) But, does not fully supersedes the 1161 result for the discrepancy 2. Future math may build further on these types of computational results. (I.e. true, verifiable facts but not human accessible.)
Other examples

**AlphaGo / AlphaGoZero:**

Core engine

Monte Carlo Tree Search (UCT, 2006) + Deep Learning

Final boost: deep learning and reinforcement learning.

Search part *and insights* may remain beyond human understanding. Update: Google’s DeepMind team is studying this issue.

Planning: We can synthesize optimal plan sequences of 1000+ steps.

Changes the notion of a “program”

A planning-enabled robot will synthesize its plans on-the-fly given its current abilities. Quite different from current pre-programmed industrial robots.
What are the consequences for human understanding of machine intelligence?