First Experiments with Neural Translation of Informal Mathematics to Formal

Qingxiang Wang   Cezary Kaliszyk   Josef Urban

University of Innsbruck

Czech Technical University in Prague

ICMS 2018
July 25, 2018

Two Obstacles to Strong AI/Reasoning for Math

1. Low reasoning power of automated reasoning methods, particularly over large complex theories
2. Lack of computer understanding of current human-level (math and exact science) knowledge

- The two are related: human-level math may require nontrivial reasoning to become fully explained. Fully explained math gives us a lot of data for training AI/TP systems.
- And we want to train AI/TP on human-level proofs too. Thus getting interesting formalization/ATP/learning feedback loops.
- In 2014 we have decided that the AI/TP systems are getting strong enough to try this. And we started to combine them with statistical translation of informal-to-formal math.

ProofWiki vs Mizar – our CICM'14 Example

File Edit View Go Bookmarks Help

1 of 1 89,84%

Example: ProofWiki vs Mizar vs Mizar-style automated proof

== Theorem ==
Let $(S, \circ)$ be an [[Definition:Algebraic Structure|algebraic structure]] that has a [[Definition:Zero Element|zero element]] $z \in S$. Then z is unique.
== Proof ==
Suppose z_1 and z_2 are both zeroes of $(S, \circ)$.
Then by the definition of [[Definition:Zero Element|zero element]]:
$z_2 \circ z_1 = z_1$ by dint of z_1 being a zero;
$z_2 \circ z_1 = z_2$ by dint of z_2 being a zero.
So $z_1 = z_2 \circ z_1 = z_2$.
So $z_1 = z_2$ and there is only one zero after all.
{{qed}}
// NB: Informal proofs are buggy!

```
Th9:  e1 is_a_left_unity_wrt o &
e2 is_a_right_unity_wrt o implies e1 = e2
proof
assume that A1:  e1 is_a_left_unity_wrt o and
A2:  e2 is_a_right_unity_wrt o;
thus e1 = o.(e1,e2) by A2,Def6 .= e2 by A1,Def5;
end;

z1 is_a_unity_wrt o & z2 is_a_unity_wrt o
implies z1 = z2 proof
assume that A1:  z1 is_a_unity_wrt o and
A2:  z2 is_a_unity_wrt o;
A3:  o.(z2,z1) = z1 by Th3,A2; ::[ATP]
A4:  o.(z2,z1) = z2 by Def 6,Def 7,A1,A3; ::[ATP]
hence z1 = z2 by Th9,A1,Def 7,A2; ::[ATP]
end;
```

Formal, Informal and Semiformal Corpora

- HOL Light and Flyspeck: some 25,000 toplevel theorems
- The Mizar Mathematical Library: some 60,000 toplevel theorems (most of them rather small lemmas), 10,000 definitions
- Coq: several large projects (Feit-Thompson theorem, ...)
- Isabelle, seL4 and the Archive of Formal Proofs
- Arxiv.org: 1M articles collected over some 20 years (not just math)
- Wikipedia: 25,000 articles in 2010 - collected over 10 years only
- Proofwiki - LaTeX but very semantic, re-invented the Mizar proof style

Our Initial Approach/Plan

- There is not yet much aligned informal/formal data
- So try first with “ambiguated” (informalized) formal corpora
- Try first with non black-box architectures such as probabilistic grammars
- Which can be easily enhanced internally by semantic pruning (e.g. type constraints)
- Develop feedback loops between training statistical parsing and theorem proving
- Start employing more sophisticated ML methods
- Progress to more complicated informal corpora/phenomena
- Both directly: ML/ATP with only cruder alignments (theorems, chapters, etc)
- And indirectly: train statistical/precise alignments across informal and formal corpora, use them to enhance our coverage
- Example: word2vec/Glove/neural learning of synonyms over Arxiv

Work Done So Far: Informalized Flyspeck

- 22000 Flyspeck theorem statements informalized
  - 72 overloaded instances like “+” for `vector_add`
  - 108 infix operators
  - forget “prefixes” `real_, int_, vector_, matrix_, complex_`, etc.
  - REAL_NEGNEG: $\forall x. - -x = x$

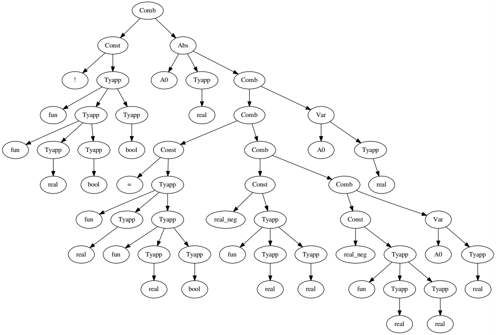
```
(Comb (Const "!" (Tyapp "fun" (Tyapp "fun" (Tyapp "real") (Tyapp "bool"))
(Tyapp "bool"))) (Abs "A0" (Tyapp "real") (Comb (Comb (Const "=" (Tyapp "fu
(Tyapp "real") (Tyapp "fun" (Tyapp "real") (Tyapp "bool")))) (Comb (Const
"real_neg" (Tyapp "fun" (Tyapp "real") (Tyapp "real"))) (Comb (Const
"real_neg" (Tyapp "fun" (Tyapp "real") (Tyapp "real"))) (Var "A0" (Tyapp
"real"))))) (Var "A0" (Tyapp "real")))))
```

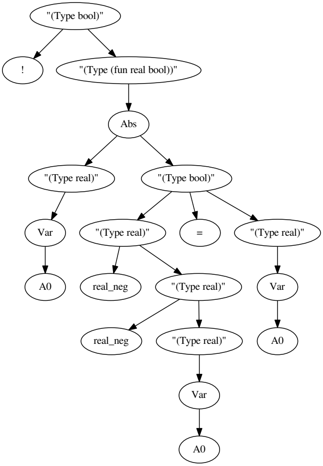
  - becomes

```
("̈(Type bool)"̈ ! ("̈(Type (fun real bool))"̈ (Abs ("̈(Type real)"̈
(Var A0)) ("̈(Type bool)"̈ ("̈(Type real)"̈ real_neg ("̈(Type real)"̈
real_neg ("̈(Type real)"̈ (Var A0)))) = ("̈(Type real)"̈ (Var A0))))))
```

- Training a probabilistic grammar (context-free, later with deeper context)
- CYK chart parser with semantic pruning (compatible types of variables)
- Using HOL Light and HolyHammer to typecheck and prove the results

Example grammars



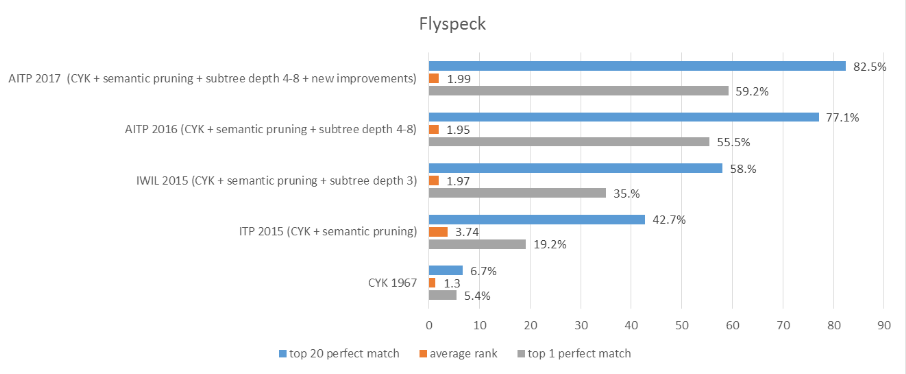


Online parsing system

- “sin (0 * x) = cos pi / 2”
- produces 16 parses
- of which 11 get type-checked by HOL Light as follows
- with all but three being proved by HOL(y)Hammer

```
sin (&0 * A0) = cos (pi / &2) where A0:real
sin (&0 * A0) = cos pi / &2 where A0:real
sin (&0 * &A0) = cos (pi / &2) where A0:num
sin (&0 * &A0) = cos pi / &2 where A0:num
sin (&(0 * A0)) = cos (pi / &2) where A0:num
sin (&(0 * A0)) = cos pi / &2 where A0:num
csin (Cx (&0 * A0)) = ccos (Cx (pi / &2)) where A0:real
csin (Cx (&0) * A0) = ccos (Cx (pi / &2)) where A0:real^2
Cx (sin (&0 * A0)) = ccos (Cx (pi / &2)) where A0:real
csin (Cx (&0 * A0)) = Cx (cos (pi / &2)) where A0:real
csin (Cx (&0) * A0) = Cx (cos (pi / &2)) where A0:real^2
```

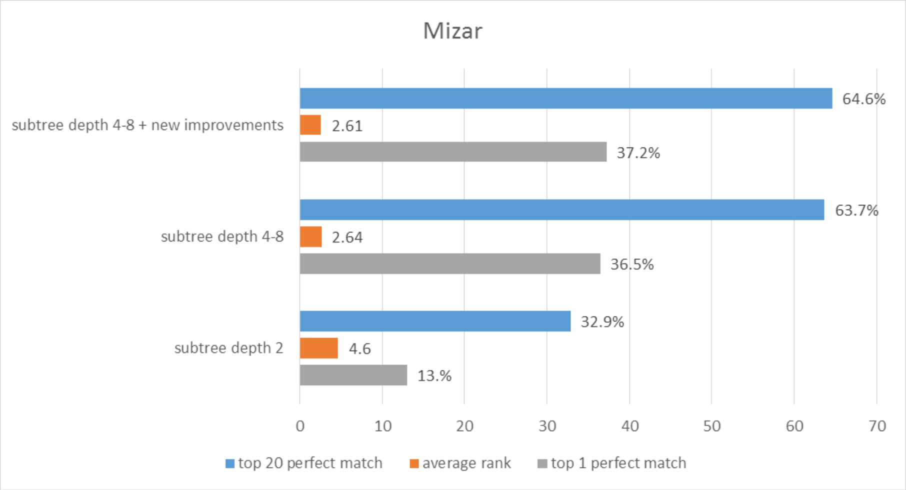
Flyspeck Progress



Tried Also for Mizar

- More natural-language features than HOL (designed by a linguist)
- Pervasive overloading
- Declarative natural-deduction proof style (re-invented in ProofWiki)
- Adjectives, dependent types, hidden arguments, synonyms
- Addressed by using two layers
  - user (pattern) layer - resolves overloading, but no hidden arguments completed, etc.
  - semantic (constructor) layer - hidden arguments computed, types resolved, ATP-ready
  - connected by ATP or a custom elaborator

First Mizar Results (100-fold Cross-validation)



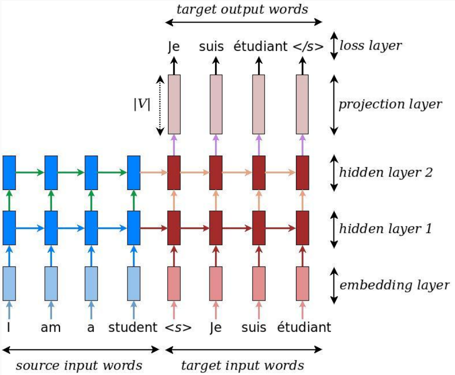
Neural Autoformalization (Wang et al., 2018)

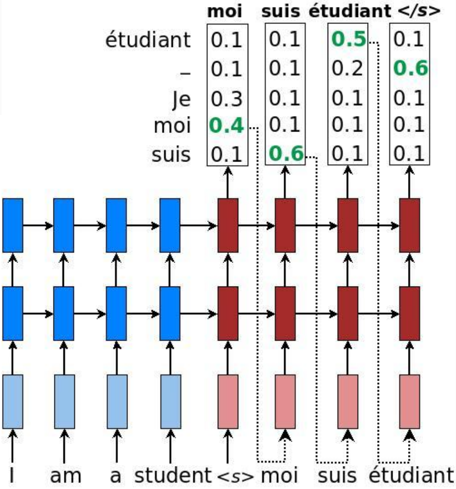
- generate about 1M Latex - Mizar pairs
- Based on Bancerek's work: journal *Formalized Mathematics* `http://fm.mizar.org/`
- train neural seq-to-seq translation models (Luong – NMT)
- evaluate on about 100k examples
- many architectures tested, some work much better than others
- very important latest invention: *attention* in the seq-to-seq models
- more data very important for neural training – our biggest bottleneck (you can help!)

Neural Autoformalization data

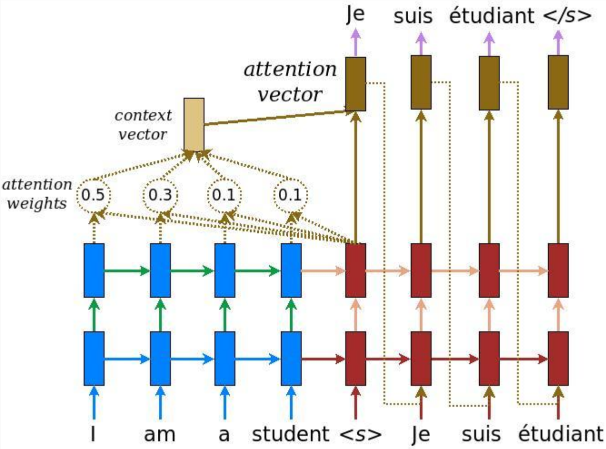
Rendered LaTeX	If $X \subseteq Y \subseteq Z$, then $X \subseteq Z$.
Mizar	`X c= Y & Y c= Z implies X c= Z;`
Tokenized Mizar	`X c= Y & Y c= Z implies X c= Z ;`
LaTeX	`If $X \subseteq Y \subseteq Z$, then $X \subseteq Z$.`
Tokenized LaTeX	`If $ X \subseteq Y \subseteq Z $ , then $ X \subseteq Z $ .`

Sequence-to-sequence models - decoder/encoder RNN





Seq2seq with Attention



Neural Autoformalization results

Parameter	Final Test Perplexity	Final Test BLEU	Identical Statements (%)	Identical No-overlap (%)
128 Units	3.06	41.1	40121 (38.12%)	6458 (13.43%)
256 Units	1.59	64.2	63433 (60.27%)	19685 (40.92%)
512 Units	1.6	**67.9**	66361 (63.05%)	21506 (44.71%)
1024 Units	**1.51**	61.6	**69179 (65.73%)**	**22978 (47.77%)**
2048 Units	2.02	60	59637 (56.66%)	16284 (33.85%)

Neural Autoformalization - Mizar to LaTeX

Parameter	Final Test Perplexity	Final Test BLEU	Identical Statements	Percentage
512 Units Bidirectional Scaled Luong	2.91	57	54320	51.61%

Coverage and Edit Instance

	Identical Statements	0	≤ 1	≤ 2
Best Model - 1024 Units	69179 (total) 22978 (no-overlap)	65.73% 47.77%	74.58% 59.91%	86.07% 70.26%
Top-5 Greedy Cover - 1024 Units - 4-Layer Bi. Res. - 512 Units - 6-Layer Adam Bi. Res. - 2048 Units	78411 (total) 28708 (no-overlap)	74.50% 59.68%	82.07% 70.85%	87.27% 78.84%
Top-10 Greedy Cover - 1024 Units - 4-Layer Bi. Res. - 512 Units - 6-Layer Adam Bi. Res. - 2048 Units - 2-Layer Adam Bi. Res. - 256 Units - 5-Layer Adam Res. - 6-Layer Adam Res. - 2-Layer Bi. Res.	80922 (total) 30426 (no-overlap)	76.89% 63.25%	83.91% 73.74%	88.60% 81.07%
Union of All 39 Models	83321 (total) 32083 (no-overlap)	79.17% 66.70%	85.57% 76.39%	89.73% 82.88%

Neural Fun – Performance after Some Training

Rendered LaTeX

Suppose s_8 is convergent and s_7 is convergent . Then $\lim(s_8+s_7) = \lim s_8 + \lim s_7$

Input LaTeX

```
Suppose $ { s _ { 8 } } $ is convergent and $ { s _ { 7 } }
$ is convergent . Then $ \mathop { \rm lim } ( { s _ { 8 }
} { + } { s _ { 7 } } ) \mathrel { = } \mathop { \rm lim }
{ s _ { 8 } } { + } \mathop { \rm lim } { s _ { 7 } } $ .
```

Correct

```
seq1 is convergent & seq2 is convergent implies lim ( seq1
+ seq2 ) = ( lim seq1 ) + ( lim seq2 ) ;
```

Snapshot-1000

```
x in dom f implies ( x * y ) * ( f | ( x | ( y | ( y | y )
) ) ) = ( x | ( y | ( y | ( y | y ) ) ) ) ) ;
```

Snapshot-2000

```
seq is summable implies seq is summable ;
```

Snapshot-3000

```
seq is convergent & lim seq = 0c implies seq = seq ;
```

Snapshot-4000

```
seq is convergent & lim seq = lim seq implies seq1 + seq2
is convergent ;
```

Snapshot-5000

```
seq1 is convergent & lim seq2 = lim seq2 implies lim_inf
seq1 = lim_inf seq2 ;
```

Snapshot-6000

```
seq is convergent & lim seq = lim seq implies seq1 + seq2
is convergent ;
```

Snapshot-7000

```
seq is convergent & seq9 is convergent implies
lim ( seq + seq9 ) = ( lim seq ) + ( lim seq9 ) ;
```

Thanks and advertisement

- To push AI methods in math and theorem proving, we organize:
- AITP – Artificial Intelligence and Theorem Proving
- April 8–12, 2019, Obergurgl, Austria, `aitp-conference.org`
- ATP/ITP/ vs AI/Machine-Learning people, Computational linguists
- Discussion-oriented and experimental