AI and Theorem Proving

Josef Urban

Czech Technical University in Prague

New Technologies in Mathematics Seminar
January 13, 2021, Harvard University

European Research Council
Established by the European Commission

Outline

Motivation, Learning vs. Reasoning

Computer Understandable (Formal) Math

Learning of Theorem Proving

Examples and Demos

High-level Reasoning Guidance: Premise Selection

Low Level Guidance of Theorem Provers

Mid-level Reasoning Guidance

More on Neural Guidance, Synthesis and Conjecturing

Autoformalization

How Do We Automate Math and Science?

- What is mathematical and scientific thinking?
- Pattern-matching, analogy, induction from examples
- Deductive reasoning
- Complicated feedback loops between induction and deduction
- Using a lot of previous knowledge - both for induction and deduction

- We need to develop such methods on computers
- Are there any large corpora suitable for nontrivial deduction?
- Yes! Large libraries of formal proofs and theories
- So let's develop strong AI on them!

What is Formal Mathematics?

- Developed thanks to the Leibniz/Russell/Frege/Hilbert/... program
- Mathematics put on formal logic foundations (*symbolic computation*)
- ... which btw. led also to the rise of computers (Turing/Church, 1930s)
- Formal math (1950/60s): combine formal foundations and the newly available computers
- De Bruijn, Milner, Trybulec, Boyer and Moore, Gordon, Huet, Paulson, ...
- Automath, LCF, Mizar, NQTHM and ACL2, HOL, Coq, Isabelle, ...
- Conceptually very simple:
- Write all your axioms and theorems so that computer understands them
- Write all your inference rules so that computer understands them
- Use the computer to check that your proofs follow the rules
- But in practice, it turns out not to be so simple
- Many approaches, still not mainstream, but big breakthroughs recently

History and Motivation for AI/TP

- Intuition vs Formal Reasoning – Poincaré vs Hilbert, Science & Method
- Turing's 1950 paper: Learning Machines, learn Chess?, undecidability??
- Lenat, Langley, etc: manually-written heuristics, learn Kepler laws,...
- Denzinger, Schulz, Goller, Fuchs – late 90's, ATP-focused:
- *Learning from Previous Proof Experience*
- My MSc (1998): Try ILP to learn rules and heuristics from IMPS/Mizar
- Since: Use large formal math (Big Proof) corpora: Mizar, Isabelle, HOL
- ... to combine/develop symbolic/statistical deductive/inductive ML/TP/AI
- ... hammer-style methods, feedback loops, internal guidance, ...
- More details – AGI'18 keynote: `https://bit.ly/3qifhg4`
- AI vs DL: Ben Goertzel's Prague talk: `https://youtu.be/Zt2HSTuGBn8`
- Big AI visions: automate/verify math, science, law, (Leibniz, McCarthy, ..)
- Practical impact: boost today's large ITP verification projects

Using Learning to Guide Theorem Proving

- **high-level**: pre-select lemmas from a large library, give them to ATPs
- **high-level**: pre-select a good ATP strategy/portfolio for a problem
- **high-level**: pre-select good *hints* for a problem, use them to guide ATPs
- **low-level**: guide every inference step of ATPs (tableau, superposition)
- **low-level**: guide every kernel step of LCF-style ITPs
- **mid-level**: guide application of tactics in ITPs
- **mid-level**: invent suitable ATP strategies for classes of problems
- **mid-level**: invent suitable conjectures for a problem
- **mid-level**: invent suitable concepts/models for problems/theories
- **proof sketches**: explore stronger/related theories to get proof ideas
- **theory exploration**: develop interesting theories by conjecturing/proving
- **feedback loops**: (dis)prove, learn from it, (dis)prove more, learn more, ...
- **autoformalization**: (semi-)automate translation from LaTeX to formal
- ...

Large AI/TP Datasets

- Mizar / MML / MPTP – since 2003
- MPTP Challenge (2006), MPTP2078 (2011), Mizar40 (2013)
- Isabelle (and AFP) – since 2005
- Flyspeck (including core HOL Light and Multivariate) – since 2012
- HOL4 – since 2014, CakeML – 2017, GRUNGE – 2019
- Coq – since 2013/2016
- ACL2 – 2014?
- Lean?, Stacks?, Arxiv?, ProofWiki?, ...

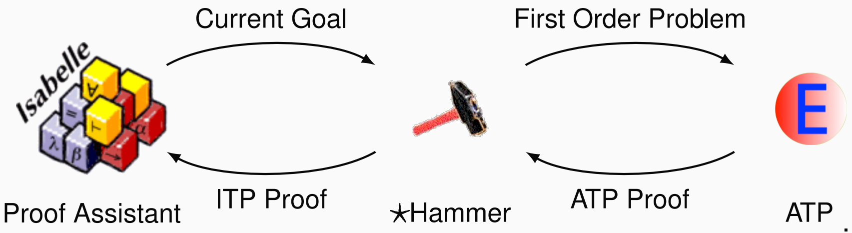
Demos

- ENIGMA/hammer proofs of Pythagoras : `https://bit.ly/2MVPAn7` (more at `http://grid01.ciirc.cvut.cz/~mptp/enigma-ex.pdf`) and simplified Carmichael `https://bit.ly/3oGBdRz`
- Hammering demo: `http://grid01.ciirc.cvut.cz/~mptp/out4.ogv`
- TacticToe on HOL4: `http://grid01.ciirc.cvut.cz/~mptp/tactictoe_demo.ogv`
- Tactician for Coq: `https://blaauwbroek.eu/papers/cicm2020/demo.mp4`, `https://coq-tactician.github.io/demo.html`
- Inf2formal over HOL Light: `http://grid01.ciirc.cvut.cz/~mptp/demo.ogv`

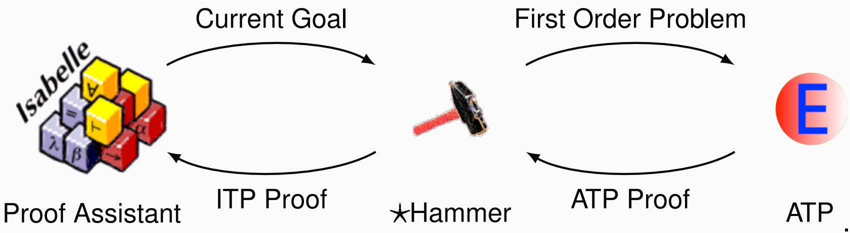
High-level ATP guidance: Premise Selection

- Early 2003: Can existing ATPs be used over the freshly translated Mizar library?
- About 80000 nontrivial math facts at that time – impossible to use them all
- Is good premise selection for proving a new conjecture possible at all?
- Or is it a mysterious power of mathematicians? (Penrose)
- Today: Premise selection is not a mysterious property of mathematicians!
- Reasonably good algorithms started to appear (more below).
- Extensive human (math) knowledge obsolete?? (cf. Watson, Debater, ..)
- Since 2004 (my PhD): many examples of nontrivial alternative proofs proposed by the AIs - in Mizar, Flyspeck, Isabelle, ..
- The premise selection algorithms see *wider* than humans

Today's AI-ATP systems (⋆-Hammers)

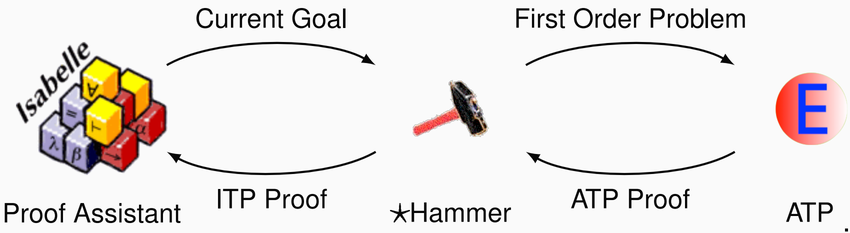


Today's AI-ATP systems (⋆-Hammers)



How much can it do?

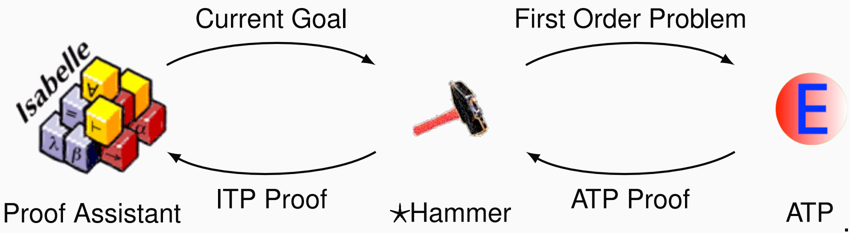
Today's AI-ATP systems (⋆-Hammers)



How much can it do?

- Mizar / MML – MizAR
- Isabelle (Auth, Jinja) – Sledgehammer
- Flyspeck (including core HOL Light and Multivariate) – HOL(y)Hammer
- HOL4 (Gauthier and Kaliszyk)
- CoqHammer (Czajka and Kaliszyk) - about 40% on Coq standard library

Today's AI-ATP systems (⋆-Hammers)



How much can it do?

- Mizar / MML – MizAR
- Isabelle (Auth, Jinja) – Sledgehammer
- Flyspeck (including core HOL Light and Multivariate) – HOL(y)Hammer
- HOL4 (Gauthier and Kaliszyk)
- CoqHammer (Czajka and Kaliszyk) - about 40% on Coq standard library

$\approx$ 40-45% success rate (close to 60% on Mizar as of 2021)

Premise Selection and Hammer Methods

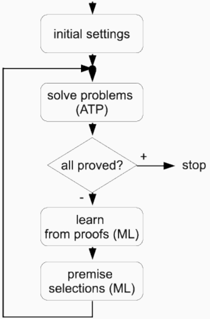
- Many syntactic features (symbols, walks in the parse trees)
- More semantic features encoding
- term matching/unification, validity in models, latent semantics (LSI)
- Distance-weighted k-nearest neighbor, SVMs, Naive Bayes
- Gradient boosted decision trees (GBDTs - XGBoost, LightGBM)
- Neural models: CNNs, RNNs/Attention/Transformers/GPT, GNNs
- As of 2020, tough competition between GBDTs, GNNs and RNNs/Transformers (and relatives)
- K-NN still very good, Olsak's logic-aware GNN probably best
- RNNs/Transformers good at stateful premise selection (Piotrowski 2019,2020)
- Ensemble methods combining the different predictors help a lot

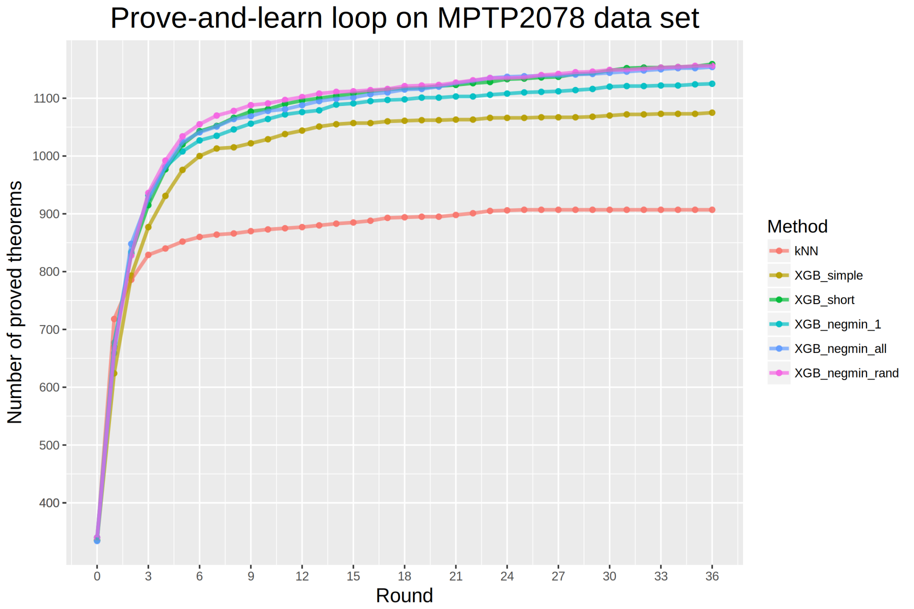
Premise Selection and Hammer Methods

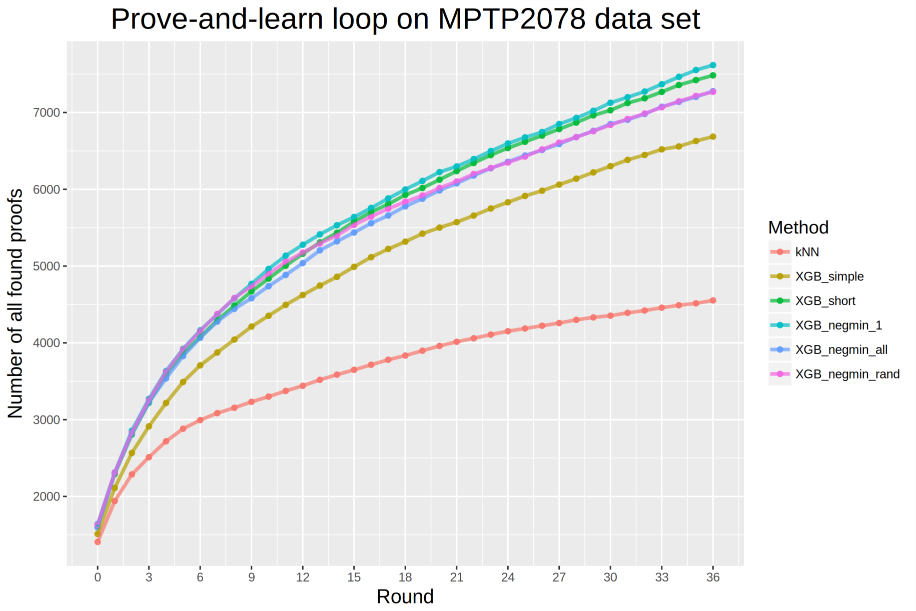
- Learning in a binary setting from many alternative proofs
- Interleaving many learning and proving runs (*MaLARea loop*) to get positives/negatives (ATPBoost - Piotrowski 2018)
- Matching and transferring concepts and theorems between libraries (Gauthier & Kaliszyk) – allows "superhammers", conjecturing, and more
- Lemmatization – extracting and considering millions of low-level lemmas and learning from their proofs
- Hammers combined with guided tactical search: TacticToe (Gauthier - HOL4) and its later relatives

High-level feedback loops – MALARea, ATPBoost

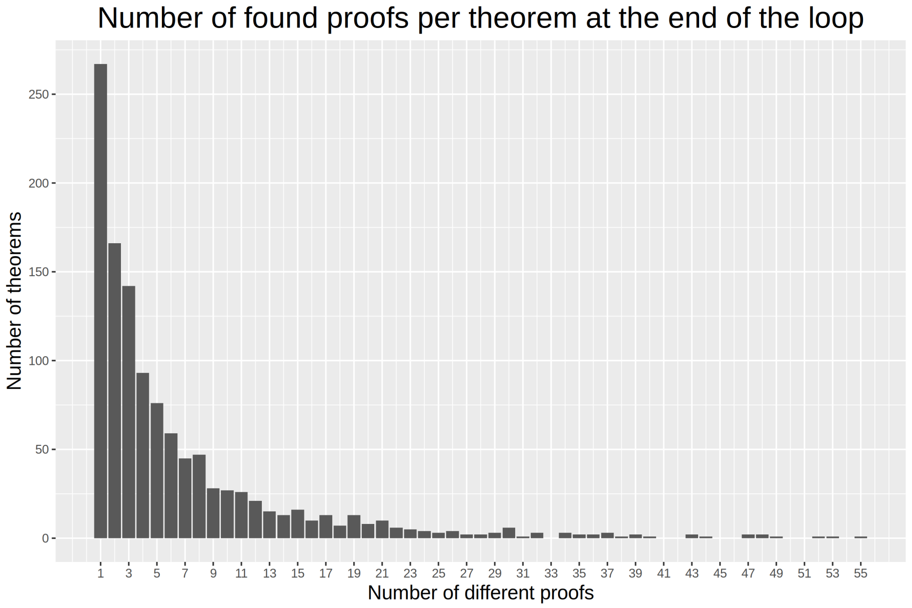
- Machine Learner for Autom. Reasoning (2006) – infinite hammering
- feedback loop interleaving ATP with learning premise selection
- both syntactic and semantic features for characterizing formulas:
- evolving set of finite (counter)models in which formulas evaluated
- winning AI/ATP benchmarks (MPTPChallenge, CASC 2008/12/13/18)
- ATPBoost (Piotrowski) - recent incarnation focusing on multiple proofs




Prove-and-learn loop on MPTP2078 data set
Number of proved theorems
1100
1000
900
800
700
600
500
400
0
3
6
9
12
15
18
21
24
27
30
33
36
Round
Method
kNN
XGB_simple
XGB_short
XGB_negmin_1
XGB_negmin_all
XGB_negmin_rand



Prove-and-learn loop on MPTP2078 data set
Number of all found proofs
7000
6000
5000
4000
3000
2000
0
3
6
9
12
15
18
21
24
27
30
33
36
Round
Method
kNN
XGB_simple
XGB_short
XGB_negmin_1
XGB_negmin_all
XGB_negmin_rand


Number of found proofs per theorem at the end of the loop
Number of theorems
Number of different proofs
0
50
100
150
200
250
1 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39 41 43 45 47 49 51 53 55

Low-level: Statistical Guidance of Connection Tableau

- learn guidance of every clausal inference in connection tableau (leanCoP)
- set of first-order clauses, *extension* and *reduction* steps
- proof finished when all branches are closed
- a lot of nondeterminism, requires backtracking
- *Iterative deepening* used in leanCoP to ensure completeness
- good for learning – the tableau compactly represents the proof state

Clauses:

$c_1 : P(x)$

$c_2 : R(x, y) \vee \neg P(x) \vee Q(y)$

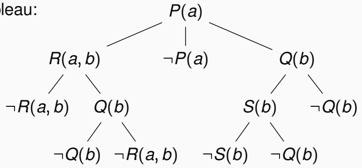
$c_3 : S(x) \vee \neg Q(b)$

$c_4 : \neg S(x) \vee \neg Q(x)$

$c_5 : \neg Q(x) \vee \neg R(a, x)$

$c_6 : \neg R(a, x) \vee Q(x)$

Closed Connection Tableau:



Statistical Guidance of Connection Tableau

- **MaLeCoP** (2011): first prototype Machine Learning Connection Prover
- extension rules chosen by naive Bayes trained on good decisions
- training examples: tableau features plus the name of the chosen clause
- initially slow: off-the-shelf learner 1000 times slower than raw leanCoP
- 20-time search shortening on the MPTP Challenge
- second version: 2015, with C. Kaliszyk
- both prover and naive Bayes in OCAML, fast indexing
- Fairly Efficient MaLeCoP = **FEMaLeCoP**
- 15% improvement over untrained leanCoP on the MPTP2078 problems
- using iterative deepening - enumerate shorter proofs before longer ones

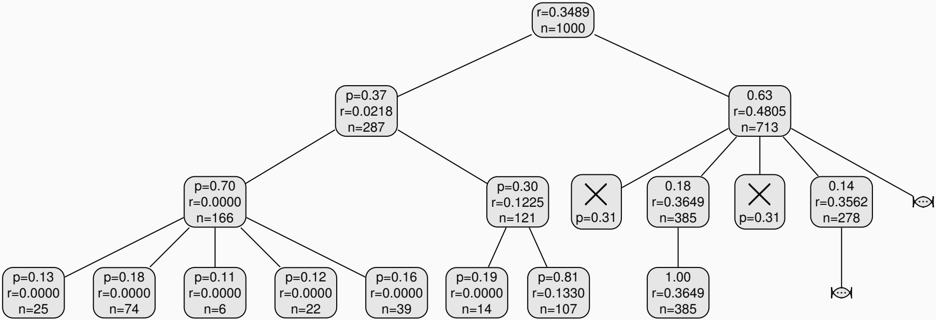
Statistical Guidance of Connection Tableau – rlCoP

- 2018: stronger learners via C interface to OCAML (boosted trees)
- remove iterative deepening, the prover can go arbitrarily deep
- added Monte-Carlo Tree Search (MCTS)
- MCTS search nodes are sequences of clause application
- a good heuristic to explore new vs exploit good nodes:

$$\frac{w_i}{n_i} + c \cdot p_i \cdot \sqrt{\frac{\ln N}{n_i}} \qquad \text{(UCT - Kocsis, Szepesvari 2006)}$$

- learning both *policy* (clause selection) and *value* (state evaluation)
- clauses represented not by names but also by features (generalize!)
- binary learning setting used: | proof state | clause features |
- mostly term walks of length 3 (trigrams), hashed into small integers
- many iterations of proving and learning

Tree Example



Statistical Guidance of Connection Tableau – rlCoP

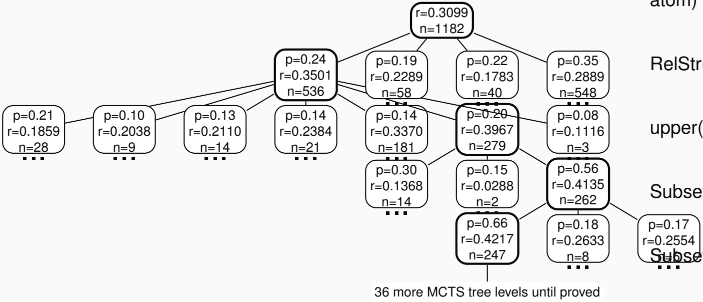
- On 32k Mizar40 problems using 200k inference limit
- nonlearning CoPs:

System	leanCoP	bare prover	rlCoP no policy/value (UCT only)
Training problems proved	10438	4184	7348
Testing problems proved	**1143**	431	804
Total problems proved	11581	4615	8152

- rlCoP with policy/value after 5 proving/learning iters on the training data
- $1624/1143 = 42.1$% improvement over leanCoP on the testing problems

Iteration	1	2	3	4	5	6	7	8
Training proved	12325	13749	14155	14363	14403	14431	14342	**14498**
Testing proved	1354	1519	1566	1595	**1624**	1586	1582	1591

More trees



(tableau starting atom)

RelStr(c1)

upper(c1)

Subset(union(c2),carrier(c1))

Subset(c2,powerset(carrier(c1)

Recent CoP Mutants: FLoP, GNN, RNN, lazyCoP

- FLoP – Finding Longer Proofs (Zombori et al, 2019)
- Curriculum Learning used for connection tableau over Robinson Arithmetic
- addition and multiplication learned perfectly from $1 * 1 = 1$
- headed towards learning algorithms/decision procedures from math data
- currently black-box, combinations with symbolic methods (ILP) our next target
- Using RNNs for better tableau encoding, prediction of actions ...
- ... even guessing (decoding) next tableau literals (Piotrowski 2020)
- plCoP (Zombori 20), GNN-CoP (Olsak 20), lazyCoP (Rawson)
- Zombori: learning new explainable Prolog actions (tactics) from proofs

ENIGMA: Guiding the Best ATPs like E Prover

- harder for learning than tableau
- the proof state are two large heaps of clauses *processed*/*unprocessed*
- 2017: ENIGMA - manual feature engineering (Jakubuv & JU 2017)
- 2017: Deep guidance (neural nets) (Loos et al. 2017)
- both learn on E's proof search traces, put classifier in E
- positive examples: given clauses used in the proof
- negative examples: given clauses not used in the proof

ENIGMA: Guiding the Best ATPs like E Prover

- ENIGMA (Jan Jakubuv 2017)
- Fast/hashed feature extraction followed by fast/sparse linear classifier
- about 80% improvement on the AIM benchmark
- Deep guidance: convolutional nets - too slow to be competitive
- ENIGMA-NG: better features and ML, gradient-boosted trees, tree NNs
- NNs made competitive in real-time, boosted trees still best
- 2020: fast GNN added (Olsak, Jakubuv), now competitive with GBDTs
- However very different: the GNN scores many clauses (context and query) simultaneously in a large graph

Feedback loop for ENIGMA on Mizar data

- Similar to rlCoP - interleave proving and learning of ENIGMA guidance
- Done on 57880 Mizar problems recently
- Serious ML-guidance breakthrough applied to the best ATPs
- Ultimately a 70% improvement over the original strategy in 2019
- From 14933 proofs to 25397 proofs (all 10s CPU - no cheating)
- Went up to 40k in more iterations and 60s time in 2020

	$\mathcal{S}$	$\mathcal{S} \odot \mathcal{M}_9^0$	$\mathcal{S} \oplus \mathcal{M}_9^0$	$\mathcal{S} \odot \mathcal{M}_9^1$	$\mathcal{S} \oplus \mathcal{M}_9^1$	$\mathcal{S} \odot \mathcal{M}_9^2$	$\mathcal{S} \oplus \mathcal{M}_9^2$	$\mathcal{S} \odot \mathcal{M}_9^3$	$\mathcal{S} \oplus \mathcal{M}_9^3$
solved	14933	16574	20366	21564	22839	22413	23467	22910	23753
$\mathcal{S}\%$	+0%	+10.5%	+35.8%	+43.8%	+52.3%	+49.4%	+56.5%	+52.8%	+58.4
$\mathcal{S}+$	+0	+4364	+6215	+7774	+8414	+8407	+8964	+8822	+9274
$\mathcal{S}-$	-0	-2723	-782	-1143	-508	-927	-430	-845	-454

	$\mathcal{S} \odot \mathcal{M}_{12}^3$	$\mathcal{S} \oplus \mathcal{M}_{12}^3$	$\mathcal{S} \odot \mathcal{M}_{16}^3$	$\mathcal{S} \oplus \mathcal{M}_{16}^3$
solved	24159	24701	25100	25397
$\mathcal{S}\%$	+61.1%	+64.8%	+68.0%	+70.0%
$\mathcal{S}+$	+9761	+10063	+10476	+10647
$\mathcal{S}-$	-535	-295	-309	-183

Neural Clause Selection in Vampire (M. Suda)

Deepire: Similar to ENIGMA:

- build a *classifier* for recognizing *good* clauses
- *good* are those that appeared in past proofs

Deepire's contributions:

- Learn from clause *derivation trees only*
  Not looking at what it says, just who its ancestors were.
- Integrate using *layered clause queues*
  A smooth improvement of the base clause selection strategy.
- Tree Neural Networks: constant work per derived clause
- A signature agnostic approach
- Delayed evaluation trick (not all derived need to be evaluated)

Preliminary Evaluation on Mizar "57880"

- Learn from 63595 proofs of 23071 problems (three 30s runs)
- Deepire solves 26217 (i.e. +4054) problems in a *single 10s run*

TacticToe: mid-level ITP Guidance (Gauthier'17,18)

- TTT learns from human and its own tactical HOL4 proofs
- No translation or reconstruction needed - native tactical proofs
- Fully integrated with HOL4 and easy to use
- Similar to rlCoP: policy/value learning for applying tactics in a state
- However much more technically challenging - a real breakthrough:
  - tactic and goal state recording
  - tactic argument abstraction
  - absolutization of tactic names
  - nontrivial evaluation issues
  - these issues have often more impact than adding better learners
- policy: which tactic/parameters to choose for a current goal?
- value: how likely is this proof state succeed?
- 66% of HOL4 toplevel proofs in 60s (better than a hammer!)
- similar recent work for Isabelle (Nagashima 2018), HOL Light (Google)

Tactician: Tactical Guidance for Coq (Blaauwbroek'20)


- Tactical guidance of Coq proofs
- Technically very challenging to do right - the Coq internals again nontrivial
- 39.3% on the Coq standard library, 56.7% in a union with CoqHammer (orthogonal)
- Fast approximate hashing for k-NN makes a lot of difference
- Speed more important than better learners
- Fully integrated with Coq, should work for any development
- User friendly, installation friendly, integration friendly and maintenance friendly
- Took several years, but could become a very common tool for Coq formalizers

Symbolic Rewriting with NNs

- Recurrent NNs with attention good at the inf2formal task
- Piotrowski 2018/19: Experiments with using RNNs for symbolic rewriting
- We can learn rewrite rules from sufficiently many data
- 80-90% success on AIM datasets, 70-99% on normalizing polynomials
- again, complements symbolic methods like ILP that suffer on big data
- in 2019 similar tasks taken up by Facebook - integration, etc.

Symbolic Rewriting Datasets

Table: Examples in the AIM data set.

Rewrite rule:	Before rewriting:	After rewriting:
`b(s(e,v1),e)=v1`	`k(b(s(e,v1),e),v0)`	`k(v1,v0)`
`o(V0,e)=V0`	`t(v0,o(v1,o(v2,e)))`	`t(v0,o(v1,v2))`

Table: Examples in the polynomial data set.

Before rewriting:	After rewriting:
`(x * (x + 1)) + 1`	`x ^ 2 + x + 1`
`(2 * y) + 1 + (y * y)`	`y ^ 2 + 2 * y + 1`
`(x + 2) * ((2 * x) + 1) + (y + 1)`	`2 * x ^ 2 + 5 * x + y + 3`

RL for Normalization and Synthesis Tasks

- Gauthier'19,20:
- Tree Neural Nets and RL (MCTS, policy/value) for:
- Guiding normalization in Robinson arithmetic
- Guiding synthesis of combinators for a given lambda expression
- Guiding synthesis of a diophantine equation characterizing a given set
- Quite encouraging results with a good curriculum (LPAR, CICM)
- Motivated by his TacticToe: argument synthesis and conjecturing is the big missing piece
- Unlike Piotrowski's RNNs/transformers, the results are series of applications of correct/explainable rules
- Gauthier's deep RL framework verifies the whole series (proof) in HOL4

RL for Normalization and Synthesis Tasks - teaser

- J. Piepenbrock (to be submitted): greatly improved RL for
- Gauthier's normalization in Robinson arithmetic
- Achieved good performance also on the polynomial normalization tasks
- Achieves performance similar to a top equational prover on the AIM problems
- Exciting: again, this is all in the verifiable/explainable proof format

More on Conjecturing in Mathematics

- Targeted: generate intermediate lemmas (cuts) for a harder conjecture
- Unrestricted (theory exploration):
- Creation of interesting conjectures based on the previous theory
- One of the most interesting activities mathematicians do (how?)
- Higher-level AI/reasoning task - can we learn it?
- If so, we have solved math:
- ... just (recursively) divide Fermat into many subtasks ...
- ... and conquer (I mean: hammer) them away

A bit of conjecturing history

- The topic goes back at least to Lenat (AM) and Fajtlowicz (Graffiti)
- Combined with automated theorem proving by Colton et al. in early 2000s (HR)
- Theory exploration for Isabelle by Johansson et al (Hipster)
- Several learning-based/neural approaches by our groups since 2015:
- Based mainly on learning analogies and informalization followed by probabilistic/neural disambiguation ...
- ... Gauthier, Kaliszyk, Chvalovsky, Piotrowski, Goertzel, Wang, Brown, JU

Conjecturing and Proof Synthesis by Neural Language models

- Karpathy'15 - RNN experiments with generating fake Math over Stacks
- I have tried to use that for formal math in 2016 but it looked weak
- GPT (-2,3) looks stronger
- Renewed experiments in 2020 on:
- All Mizar articles, stripped of comments and concatenated together (78M)
- Articles with added context/disambiguation (156M) (types, names, thesis)
- TPTP proofs of 28271 Mizar/MPTP theorems by E/ENIGMA (658M)
- Just the conjecture and premises needed for the 28271 proofs printed in prefix notation
- Quite interesting results, server for Mizar authors
- Quickly taken up by others on HOL, Isabelle, MetaMath ...

Can you find the flaw(s) in this fake GPT-2 proof?

Applications Places

emacs@dell

File Edit Options Buffers Tools Index Mizar Hide/Show Help

```
:: generated theorem with "proof"
theorem Th23: :: STIRL2_1:23
for X, Y being finite set st not X is empty & X c= Y
& card X = card Y holds X = Y
proof
  let X, Y be finite set ;
:: thesis: not X is empty & X c= Y & card X = card Y implies X = Y
  assume that
  A1: not X is empty and A2: X c= Y and A3: card X = card Y ;
:: thesis: X = Y
  card (Y \ X) = (card Y) - (card X) by A1, A3, CARD_2:44;
  then A4: card (Y \ X) = ((card Y) - 1) - (card X) by CARD_1:30;
  X = Y \ X by A2, A3, Th22;
  hence X = Y by A4, XBOOLE_0:def_10;
:: thesis: verum
end;
```

-:--- **card_tst.miz** 99% L2131 (Mizar Errors:13 hs Undo-Tree)

Figure: Fake full declarative GPT-2 “proof” - typechecks!

Mizar autocompletion server in action

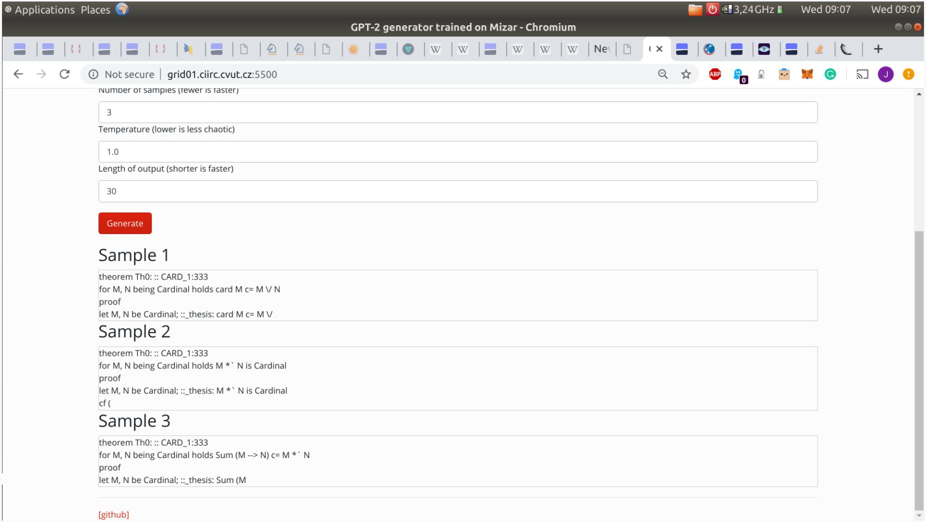


Figure: MGG - Mizar Gibberish Generator.

Proving the conditioned completions - MizAR hammer

Applications Places | Wed 14:42 | Wed 14:42

emacs@dell

File Edit Options Buffers Tools Index Mizar Hide/Show Help

Save Undo

```
begin

for M, N being Cardinal holds card M c= M \/ N by XBOOLE_1:7,CARD_3:44,CARD_1:7,CARD_1:3; :: [ATP details]

for X, Y being finite set st not X is empty & X c= Y & card X = card Y holds X = Y by CARD_FIN:1; :: [ATP details]

for M, N being Cardinal holds
( M in N iff card M c= N ) by Unsolved; :: [ATP details]

for M, N being Cardinal holds
( M in N iff card M in N ) by CARD_3:44,CARD_1:9; :: [ATP details]

for M, N being Cardinal holds Sum (M --> N) = M *` N by CARD_2:65; :: [ATP details]


for M, N being Cardinal holds M /\ (union N) in N by Unsolved; :: [ATP details]


for M, N being Cardinal holds M *` N = N *` M by ATP-Unsolved; :: [ATP details]
```

-:--- **card_tst.miz** 3% L47 (Mizar Errors:2 hs Undo-Tree)

Wrote /home/urban/mizwrk/7.13.01_4.181.1147/tst8/card_tst.miz

A correct conjecture that was too hard to prove

- Kinyon and Stanovsky (algebraists) confirmed that this cut is valid:

```
theorem Th10: :: GROUPP_1:10
for G being finite Group for N being normal Subgroup of G st
N is Subgroup of center G & G ./. N is cyclic holds G is commutative

The generalization that avoids finiteness:

for G being Group for N being normal Subgroup of G st
N is Subgroup of center G & G ./. N is cyclic holds G is commutative
```

Gibberish Generator Provoking Algebraists

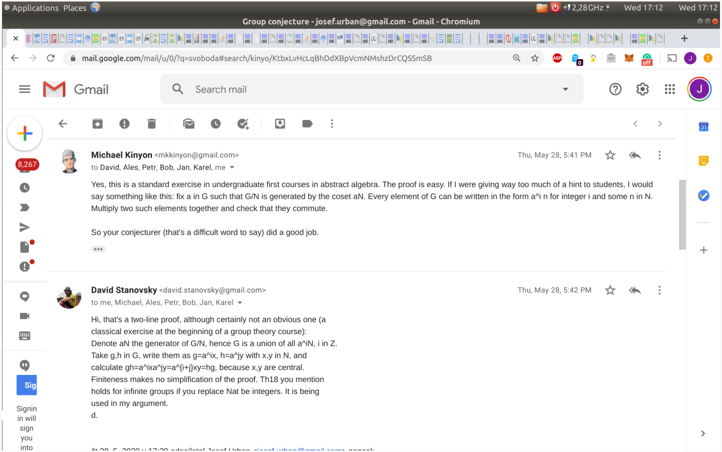


Figure: First successes in making mathematicians comment on AI.

More cuts

- In total 33100 in this experiment
- Ca 9k proved by trained ENIGMA
- Some are clearly false, yet quite natural to ask:

```
theorem :: SINCOS10:17
sec is increasing on [0, pi/2)

leads to conjecturing the following:

Every differentiable function is increasing.
```

Neural Autoformalization (Wang et al., 2018)

- generate ca 1M Latex/Mizar pairs based on Bancerek's work
- 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!)
- Recent addition: unsupervised methods (Lample et all 2018) – no need for aligned data!

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 $ .`

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 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 ) ;`

Unsupervised NMT Fun on Short Formulas

```
len <* a *> = 1 ;                   len <* a *> = 1 ;
assume i < len q ;                  i < len q ;
len <* q *> = 1 ;                   len <* q *> = 1 ;
s = apply ( v2 , v1 ast t ) ;       s = apply ( v2 , v1 ) . t ;
s . ( i + 1 ) = tt . ( i + 1 )      s . ( i + 1 ) = tau1 . ( i + 1 )
1 + j <= len v2 ;                   1 + j <= len v2 ;
1 + j + 0 <= len v2 + 1 ;           1 + j + 0 <= len v2 + 1 ;
let i be Nat ;                      i is_at_least_length_of p ;
assume v is_applicable_to t ;       not v is applicable ;
let t be type of T ;                t is_orientedpath_of v1 , v2 , T ;
a ast t in downarrow t ;            a *' in downarrow t ;
t9 in types a ;                     t `2 in types a ;
a ast t <= t ;                      a *' <= t ;
A is_applicable_to t ;              A is applicable ;
Carrier ( f ) c= B                  support ppf n c= B
u in B or u in { v } ;              u in B or u in { v } ;
F . w in w & F . w in I ;           F . w in F & F . w in I ;
GG . y in rng HH ;                  G0 . y in rng ( H1 ./. y ) ;
a * L = Z_ZeroLC ( V ) ;            a * L = ZeroLC ( V ) ;
not u in { v } ;                    u >> v ;
u <> v ;                            u <> v ;
v - w = v1 - w1 ;                   vw = v1 - w1 ;
v + w = v1 + w1 ;                   v + w = v1 + w1 ;
x in A & y in A ;                   assume [ x , y ] in A ;
```

Acknowledgments

- Prague Automated Reasoning Group `http://arg.ciirc.cvut.cz/`:
  - Jan Jakubuv, Chad Brown, Martin Suda, Karel Chvalovsky, Bob Veroff, Zar Goertzel, Bartosz Piotrowski, Lasse Blaauwbroek, Martin Smolik, Jiri Vyskocil, Petr Pudlak, David Stanovsky, Krystof Hoder, ...
- HOL(y)Hammer group in Innsbruck:
  - Cezary Kaliszyk, Thibault Gauthier, Michael Faerber, Yutaka Nagashima, Shawn Wang
- ATP and ITP people:
  - Stephan Schulz, Geoff Sutcliffe, Andrej Voronkov, Kostya Korovin, Larry Paulson, Jasmin Blanchette, John Harrison, Tom Hales, Tobias Nipkow, Andrzej Trybulec, Piotr Rudnicki, Adam Pease, ...
- Learning2Reason people at Radboud University Nijmegen:
  - Herman Geuvers, Tom Heskes, Daniel Kuehlwein, Evgeni Tsivtsivadze, ...
- Google Research: Christian Szegedy, Geoffrey Irving, Alex Alemi, Francois Chollet, Sarah Loos
- ... and many more ...
- Funding: Marie-Curie, NWO, ERC

Some General and Hammer/Tactical References

- J. C. Blanchette, C. Kaliszyk, L. C. Paulson, J. Urban: Hammering towards QED. J. Formalized Reasoning 9(1): 101-148 (2016)
- Cezary Kaliszyk, Josef Urban: Learning-Assisted Automated Reasoning with Flyspeck. J. Autom. Reason. 53(2): 173-213 (2014)
- Cezary Kaliszyk, Josef Urban: MizAR 40 for Mizar 40. J. Autom. Reason. 55(3): 245-256 (2015)
- Cezary Kaliszyk, Josef Urban: Learning-assisted theorem proving with millions of lemmas. J. Symb. Comput. 69: 109-128 (2015)
- Jasmin Christian Blanchette, David Greenaway, Cezary Kaliszyk, Daniel Kühlwein, Josef Urban: A Learning-Based Fact Selector for Isabelle/HOL. J. Autom. Reason. 57(3): 219-244 (2016)
- Bartosz Piotrowski, Josef Urban: ATPboost: Learning Premise Selection in Binary Setting with ATP Feedback. IJCAR 2018: 566-574
- T. Gauthier, C. Kaliszyk, J. Urban, R. Kumar, M. Norrish: Learning to Prove with Tactics. CoRR abs/1804.00596 (2018).
- Lasse Blaauwbroek, Josef Urban, Herman Geuvers: Tactic Learning and Proving for the Coq Proof Assistant. LPAR 2020: 138-150
- Lasse Blaauwbroek, Josef Urban, Herman Geuvers: The Tactician (extended version): A Seamless, Interactive Tactic Learner and Prover for Coq. CoRR abs/2008.00120 (2020)
- L. Czajka, C. Kaliszyk: Hammer for Coq: Automation for Dependent Type Theory. J. Autom. Reasoning 61(1-4): 423-453 (2018)
- G. Irving, C. Szegedy, A. Alemi, N. Eén, F. Chollet, J. Urban: DeepMath - Deep Sequence Models for Premise Selection. NIPS 2016: 2235-2243
- C. Kaliszyk, J. Urban, J. Vyskocil: Efficient Semantic Features for Automated Reasoning over Large Theories. IJCAI 2015: 3084-3090
- J. Urban, G. Sutcliffe, P. Pudlák, J. Vyskocil: MaLARea SG1- Machine Learner for Automated Reasoning with Semantic Guidance. IJCAR 2008: 441-456
- J. Urban, J. Vyskocil: Theorem Proving in Large Formal Mathematics as an Emerging AI Field. LNCS 7788, 240-257, 2013.

Some References on E/ENIGMA, CoPs and Related

- Stephan Schulz: System Description: E 1.8. LPAR 2013: 735-743
- S. Schulz, Simon Cruanes, Petar Vukmirovic: Faster, Higher, Stronger: E 2.3. CADE 2019: 495-507
- J. Jakubuv, J. Urban: Extending E Prover with Similarity Based Clause Selection Strategies. CICM 2016: 151-156
- J. Jakubuv,J. Urban: ENIGMA: Efficient Learning-Based Inference Guiding Machine.CICM 2017:292-302
- Cezary Kaliszyk, Josef Urban, Henryk Michalewski, Miroslav Olsák: Reinforcement Learning of Theorem Proving. NeurIPS 2018: 8836-8847
- Zarathustra Goertzel, Jan Jakubuv, Stephan Schulz, Josef Urban: ProofWatch: Watchlist Guidance for Large Theories in E. ITP 2018: 270-288
- S. M. Loos, G. Irving, C. Szegedy, C. Kaliszyk: Deep Network Guided Proof Search. LPAR 2017: 85-105
- Karel Chvalovský, Jan Jakubuv, Martin Suda, Josef Urban: ENIGMA-NG: Efficient Neural and Gradient-Boosted Inference Guidance for E. CADE 2019: 197-215
- Jan Jakubuv, Josef Urban: Hammering Mizar by Learning Clause Guidance. ITP 2019: 34:1-34:8
- Zarathustra Goertzel, Jan Jakubuv, Josef Urban: ENIGMAWatch: ProofWatch Meets ENIGMA. TABLEAUX 2019: 374-388
- Zarathustra Amadeus Goertzel: Make E Smart Again (Short Paper). IJCAR (2) 2020: 408-415
- Jan Jakubuv, Karel Chvalovský, Miroslav Olsák, Bartosz Piotrowski, Martin Suda, Josef Urban: ENIGMA Anonymous: Symbol-Independent Inference Guiding Machine. IJCAR (2) 2020: 448-463
- Zsolt Zombori, Adrián Csiszárik, Henryk Michalewski, Cezary Kaliszyk, Josef Urban: Towards Finding Longer Proofs. CoRR abs/1905.13100 (2019)
- Zsolt Zombori, Josef Urban, Chad E. Brown: Prolog Technology Reinforcement Learning Prover - (System Description). IJCAR (2) 2020: 489-507
- Miroslav Olsák, Cezary Kaliszyk, Josef Urban: Property Invariant Embedding for Automated Reasoning. ECAI 2020: 1395-1402

Some Conjecturing References

- Douglas Bruce Lenat. *AM: An Artificial Intelligence Approach to Discovery in Mathematics as Heuristic Search*. PhD thesis, Stanford, 1976.
- Siemion Fajtlowicz. On conjectures of Graffiti. *Annals of Discrete Mathematics*, 72(1–3):113–118, 1988.
- Simon Colton. *Automated Theory Formation in Pure Mathematics*. Distinguished Dissertations. Springer London, 2012.
- Moa Johansson, Dan Rosén, Nicholas Smallbone, and Koen Claessen. Hipster: Integrating theory exploration in a proof assistant. In *CICM 2014*, pages 108–122, 2014.
- Thibault Gauthier, Cezary Kaliszyk, and Josef Urban. Initial experiments with statistical conjecturing over large formal corpora. In *CICM'16 WiP Proceedings*, pages 219–228, 2016.
- Thibault Gauthier, Cezary Kaliszyk: Sharing HOL4 and HOL Light Proof Knowledge. LPAR 2015: 372-386
- Thibault Gauthier. Deep reinforcement learning in HOL4. *CoRR*, abs/1910.11797, 2019.
- Chad E. Brown and Thibault Gauthier. Self-learned formula synthesis in set theory. *CoRR*, abs/1912.01525, 2019.
- Bartosz Piotrowski, Josef Urban, Chad E. Brown, Cezary Kaliszyk: Can Neural Networks Learn Symbolic Rewriting? AITP 2019, CoRR abs/1911.04873 (2019)
- Zarathustra Goertzel and Josef Urban. Usefulness of Lemmas via Graph Neural Networks (Extende Abstract). AITP 2019.
- Karel Chvalovský, Thibault Gauthier and Josef Urban: First Experiments with Data Driven Conjecturing (Extended Abstract). AITP 2019.
- Thibault Gauthier: Deep Reinforcement Learning for Synthesizing Functions in Higher-Order Logic. LPAR 2020: 230-248
- Bartosz Piotrowski, Josef Urban: Guiding Inferences in Connection Tableau by Recurrent Neural Networks. CICM 2020: 309-314
- Josef Urban, Jan Jakubuv: First Neural Conjecturing Datasets and Experiments. CICM 2020: 315-323

References on PCFG and Neural Autoformalization

- Cezary Kaliszyk, Josef Urban, Jirí Vyskocil: Learning to Parse on Aligned Corpora (Rough Diamond). ITP 2015: 227-233
- Cezary Kaliszyk, Josef Urban, Jirí Vyskocil, Herman Geuvers: Developing Corpus-Based Translation Methods between Informal and Formal Mathematics: Project Description. CICM 2014: 435-439
- C. Kaliszyk, J. Urban, J. Vyskocil: Automating Formalization by Statistical and Semantic Parsing of Mathematics. ITP 2017: 12-27
- Cezary Kaliszyk, Josef Urban, Jirí Vyskocil: System Description: Statistical Parsing of Informalized Mizar Formulas. SYNASC 2017: 169-172
- Q. Wang, C. Kaliszyk, J. Urban: First Experiments with Neural Translation of Informal to Formal Mathematics. CICM 2018: 255-270
- Qingxiang Wang, Chad E. Brown, Cezary Kaliszyk, Josef Urban: Exploration of neural machine translation in autoformalization of mathematics in Mizar. CPP 2020: 85-98

Thanks and Advertisement

- Thanks for your attention!
- AITP – Artificial Intelligence and Theorem Proving
- September 5–10, 2021, Aussois, France, `aitp-conference.org`
- ATP/ITP/Math vs AI/Machine-Learning people, Computational linguists
- Discussion-oriented and experimental - submit a talk abstract!
- Grown to 80 people in 2019
- Will be hybrid in 2021 as in 2020