Useful Lemmas in E ATP Proofs

Zarathustra Goertzel and Josef Urban

Czech Technical University in Prague

AITP'19

Outline of talk

- What are lemmas and why do they matter?
- Quantifying lemma usefulness.
- Machine learning to identify lemmas.
- Conclusion.

Lemmas

Lemmas are:

- **True statements**
- **Intermediate results**
- **Sometimes used in multiple theorems**

Why seek lemmas?

- **ATPs struggle to find long proofs.**
- **Conjecturing new (interesting) results.**
- **Concise presentations of proofs.**

Lemmas as Cuts

Given axiom set Γ and conjecture C, we want to prove $\Gamma \vdash C$.

We call L a lemma if the following holds:

$$\frac{\Gamma \vdash L \qquad \Gamma, L \vdash C}{\Gamma \vdash C}$$

*** This doesn't require L be a "useful lemma".**

Lemmas via Excluded Middle

E is a refutational theorem prover and tries to derive a contradiction: $\Gamma, \neg C \vdash \bot$.

Therefore the problem can be broken into two sub-problems:

$$\frac{\Gamma, L \vdash C \qquad \Gamma, \neg L \vdash C}{\Gamma, (L \vee \neg L) \vdash C}$$

Lemma Usefulness:
Proof Shortening Ratio

$$psr(L, \Gamma, C) = \frac{|\Gamma, L \vdash C| + |\Gamma, \neg L \vdash C|}{|\Gamma \vdash C|}$$

If the two sub-problems can be solved (by E) with psr(L, Γ, C) < 1, L can be said to be a useful lemma.

Dataset: Built From E Proofs

- **E's a saturation-based refutational ATP.**
- **Goal: Prove conjecture from premises.**
- **E has two sets of clauses:**
  - *Processed* clauses P (initially empty)
  - *Unprocessed* clauses U (Negated Conjecture and Premises)
- **Given Clause Loop:**
  - Select '*given clause*' g to add to P
  - Apply *inference rules* to g and all clauses in P
  - Process new clauses. Add non-trivial and non-redundant ones to U.
- **Proof search succeeds when empty clause is inferred.**
- **Proof consists of given clauses.**

Down and Dirty with the Datset

- 3161 CNF problems from Mizar 40 dataset
- Proved by single E strategy
- For each clause L_i^P of proof P, solve both sub-problems.
- 230528 clauses in total

Lemma Stats

Of the 230528 clauses:

- 98472 are axioms and negated conjectures.
- 87161 are anti-useful lemmas
- 44895 are useful lemmas
- 154 have psr(L, Γ, C) = 1

Lemma Stats

- **Best lemma's psr: 0.0036 (275 times faster)**
- **Worst lemma: 77 times slower**
- **Number of lemmas under 0.1: 1509**

Lemma Classification

Why?

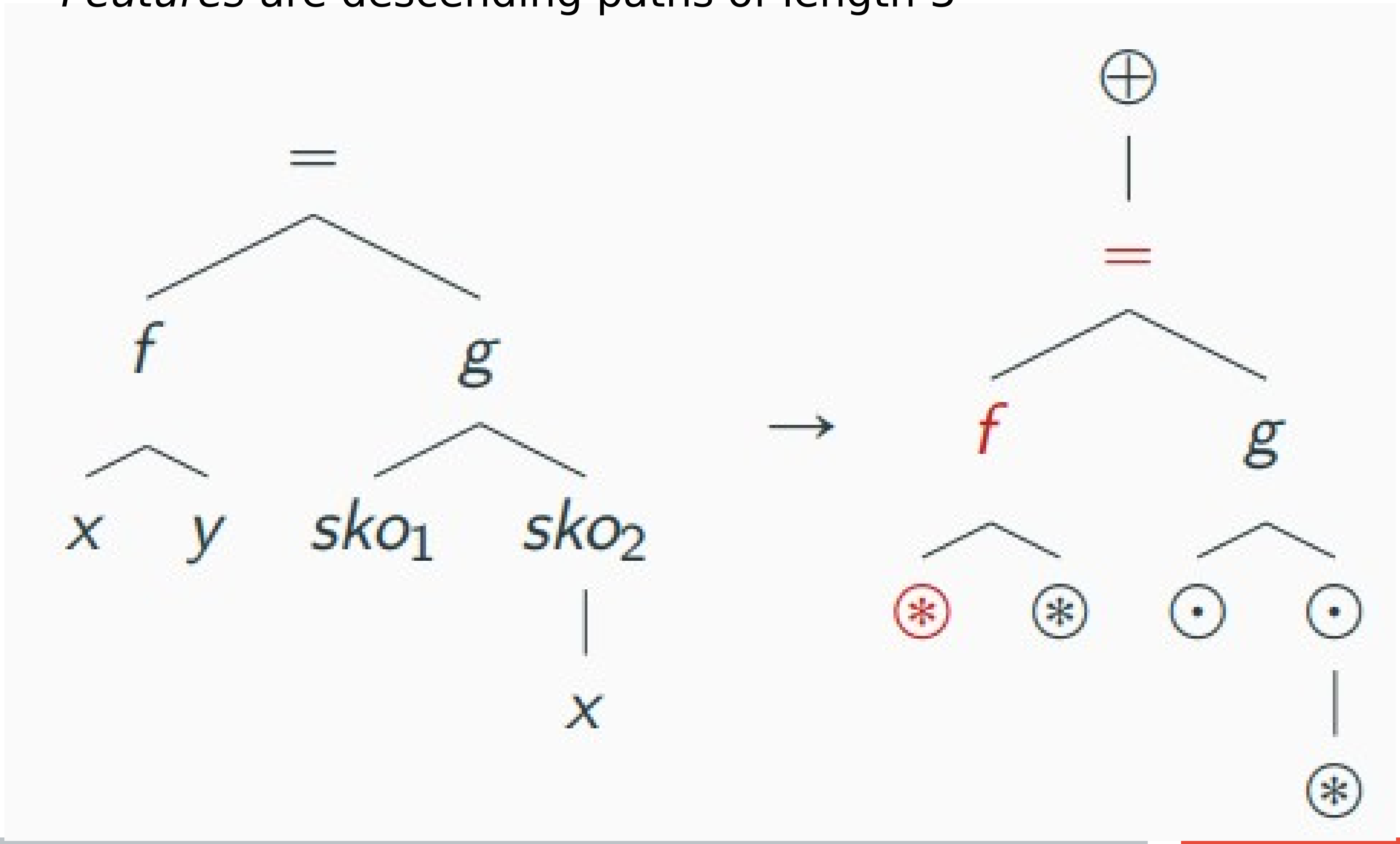
- **To gauge the difficulty of the dataset**
- **Clear yes/no results compared to regression**

Possible use-cases:

- **Proof compression for E inference guidance**
- **Analyze incomplete proof-search to look for lemmas**

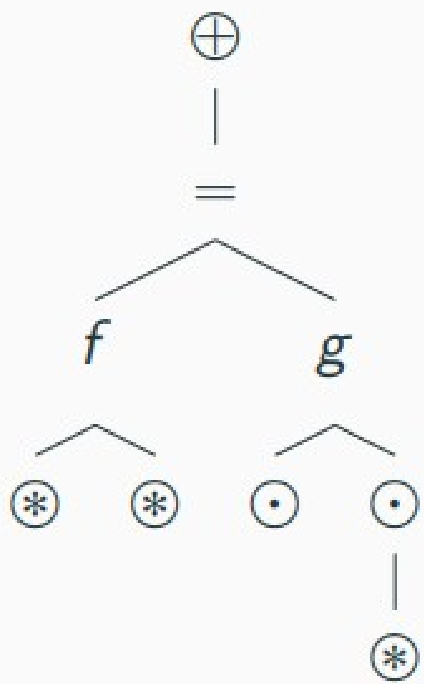
Clauses ⟶ Vectors

- Treat clause as tree. Abstract vars and skolem symbols
- *Features* are descending paths of length 3



Clauses → Vectors

Enumerate features (→ R^|Features| vector space)
Count features in a clause for its vector

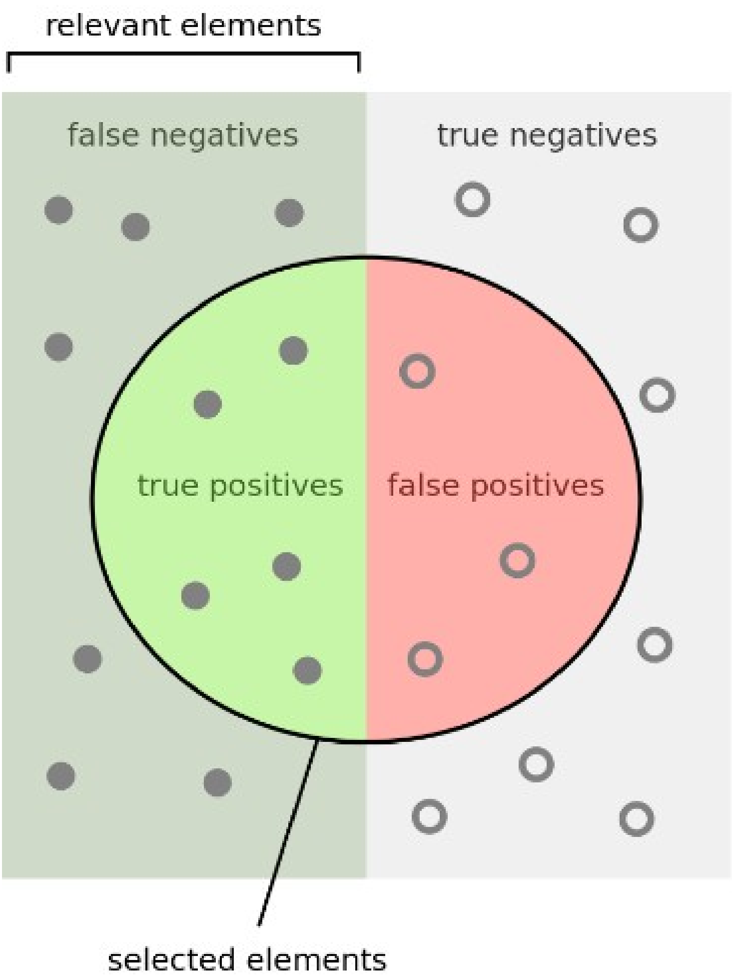


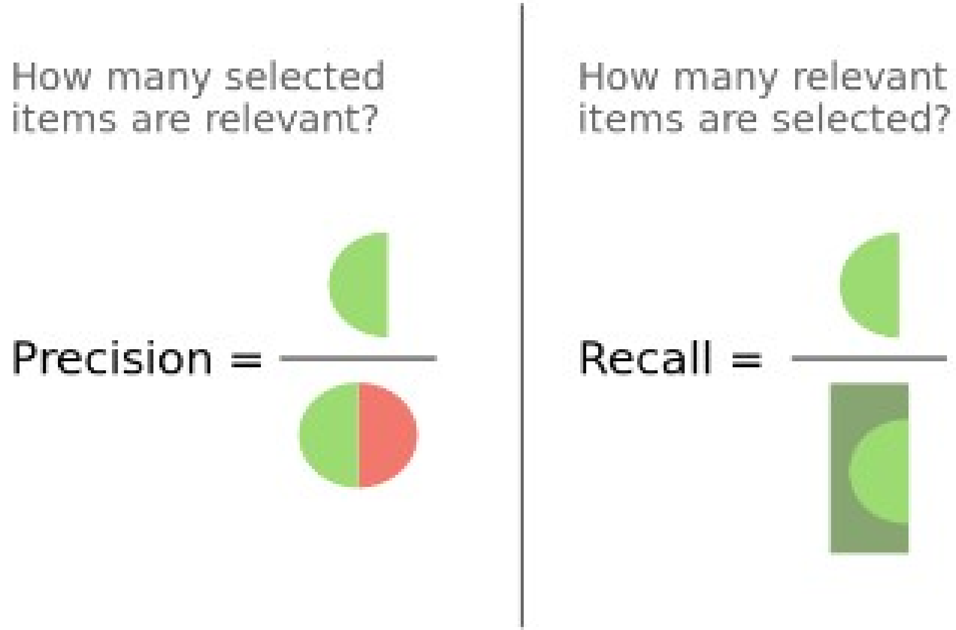
#	feature	count
1	$(\oplus,=,\mathrm{a})$	0
⋮	⋮	⋮
11	$(\oplus,=,\mathrm{f})$	1
12	$(\oplus,=,\mathrm{g})$	1
13	$(=,\mathrm{f},\circledast)$	2
14	$(=,\mathrm{g},\odot)$	2
15	$(\mathrm{g},\odot,\circledast)$	1
⋮	⋮	⋮

ML Methods

- **Support Vector Machine Classifier (SVC) from scikit-learn**
- **XGBoost: gradient boosted random decision forest:**
  - SVC and XGBoost use |Clause ++ Conjecture| Enigma features.
- **Graph Attention Networks (GAT):**
  - Assign labels or numbers to nodes via the graph structure.
  - At each level, a node’s features depend on its neighbors.
  - Drawback: graph adjacency matrix, large memory consumption
  - Question: Will the proof-graph structure help identify lemmas?

Results





F score

$$2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

Images courtesy of https://en.wikipedia.org/wiki/F1_score

Results

	F-score	Precision	Recall	Accuracy
SVC	0.53	0.45	0.64	0.74
GAT	0.55	0.45	0.72	0.55
XGBoost	0.68	0.65	0.72	0.77

Results are on a 10% test set.

Precision and Recall are with respect to useful lemmas.

Conclusions

- GAT appears not to scale, and the proof-graph is not effectively utilized.
- XGBoost is cheap to train and sufficiently effective as to be used in further experiments with E.

Todo:

- Learn more semantic features
- Work on generating lemmas