Gleaning Relational Information from Biomedical Text

Mark Goadrich, Louis Oliphant and Jude Shavlik
Department of Computer Sciences
University of Wisconsin - Madison

Abstract

Recently, biomedical journal articles have been a major source of interest in the Information Extraction (IE) community for a number of reasons: the amount of data available is enormous; the objects, proteins and genes, do not have standard naming conventions; and there is interest from biomedical practitioners to quickly find relevant relationships between these objects. We pursue IE from a machine learning perspective - in particular, by using the relational approach of Inductive Logic Programming (ILP). Given a set of Medline journal abstracts manually tagged with biological relationships, our goal is to learn a theory that extracts only these relations from a set of abstracts and performs well on unseen abstracts. We have developed Gleaner, a fast parallel ensemble-based ILP algorithm, which has performed favorably in comparison to other standard ILP approaches when applied to IE in biomedical domains.

References

Aleph
http://web.comlab.ox.ac.uk/oucl/research/areas/machlearn/Aleph/

Gene Ontology (GO)
http://www.geneontology.org/

Medical Subject Headings (MeSH)
http://www.nlm.nih.gov/mesh/meshhome.html

Medline
http://www.ncbi.nlm.nih.gov/entrez/query.fcgi

Online Medical Dictionary
http://cancerweb.ncl.ac.uk/omd/

Our Protein Localization Dataset
ftp://ftp.cs.wisc.edu/machine-learning/shavlik-group/datasets/IE-protein-location

Computation and Informatics in Biology and Medicine

We gratefully acknowledge the funding from NLM Grants 5T15LM007359 and 1R01LM07050, DARPA Grant F30602-01-2-0571, and Air Force Grant F30602-01-2-0571. We would like to thank Ines Dutra, Vitor Santos Costa, the UW Condor Group, Soumya Ray, Mark Craven, Marios Skounakis, David Page, Jesse Davis and Ameet Soni for their helpful comments on this research.

Excerpt from Medline Abstract 9121474

> ... Using GSP1, encoding the yeast Ran, as bait, we isolated YRB2. YRB2 encodes a protein containing a Ran-binding motif similar to that found in Yrb1p and Nup2p. Yrb1p is located in the cytosol whereas Nup2p is nuclear. Similar to Yrb1p, Yrb2p bound to GTP-Gsp1p but not to GDP-Gsp1p and enhanced the GTPase-activating activity of Rna1p. However, unlike Yrb1p, Yrb2p did not inhibit the nucleotide-releasing activity of Prp20p. ...

Our data comes from Medline abstracts, a large on-line collection of biomedical journal articles. Positive examples of interesting relations between biomedical objects are then labeled, such as the localization of two yeast proteins shown above.

For our information-extraction task, we construct background knowledge in first-order logic from many sources: we use the Sundance parser to find the sentence structure, then incorporate semantic properties from the MeSH and Gene Ontology biomedical dictionaries as well as lexical properties such as "alphanumeric" and "capitalized," and finally calculated statistical word frequency on the training set.

Translation from Text to Logical Prolog Notation

Background Knowledge	Some Prolog Facts Created

Sentence Structure	`sentence(ab9121474_sen6).`
	`phrase(ab9121474_sen6_ph0).`
	`phrase(ab9121474_sen6_ph1).`
	`word(ab9121474_sen6_ph0_w1).`
	`word(ab9121474_sen6_ph1 w2).`
	`word(ab9121474_sen6_ph1_w3).`
	`phrase_child(ab9121474_sen6_ph0, ab9121474_sen6_ph0_w1).`
	`word_next(ab9121474_sen6_ph0_w1, ab9121474_sen6_ph0_w2).`
	`` word_ID_to_string(ab9121474_sen6_ph1_w1, `YRB1p'). ``
	`target_arg1_before_target_arg2(ab9121474_sen6).`

Part Of Speech	`np_segment(ab9121474_sen6_ph0).`
	`vp_segment(ab9121474_sen6_ph1).`
	`unk(ab9121474_sen6_ph0_w0).`
	`cop(ab9121474_sen6_ph1_w1).`
	`v(ab9121474_sen6_ph1_w2).`

Medical Ontologies	`` phrase_contains_mesh_term(ab9121474_sen6_ph3, `cytosol'). ``
	`` phrase_contains_medDict_term(ab9121474_sen6_ph3, `cytosol'). ``
	`` phrase_contains_go_term(ab9121474_sen6_ph3, `cytosol'). ``

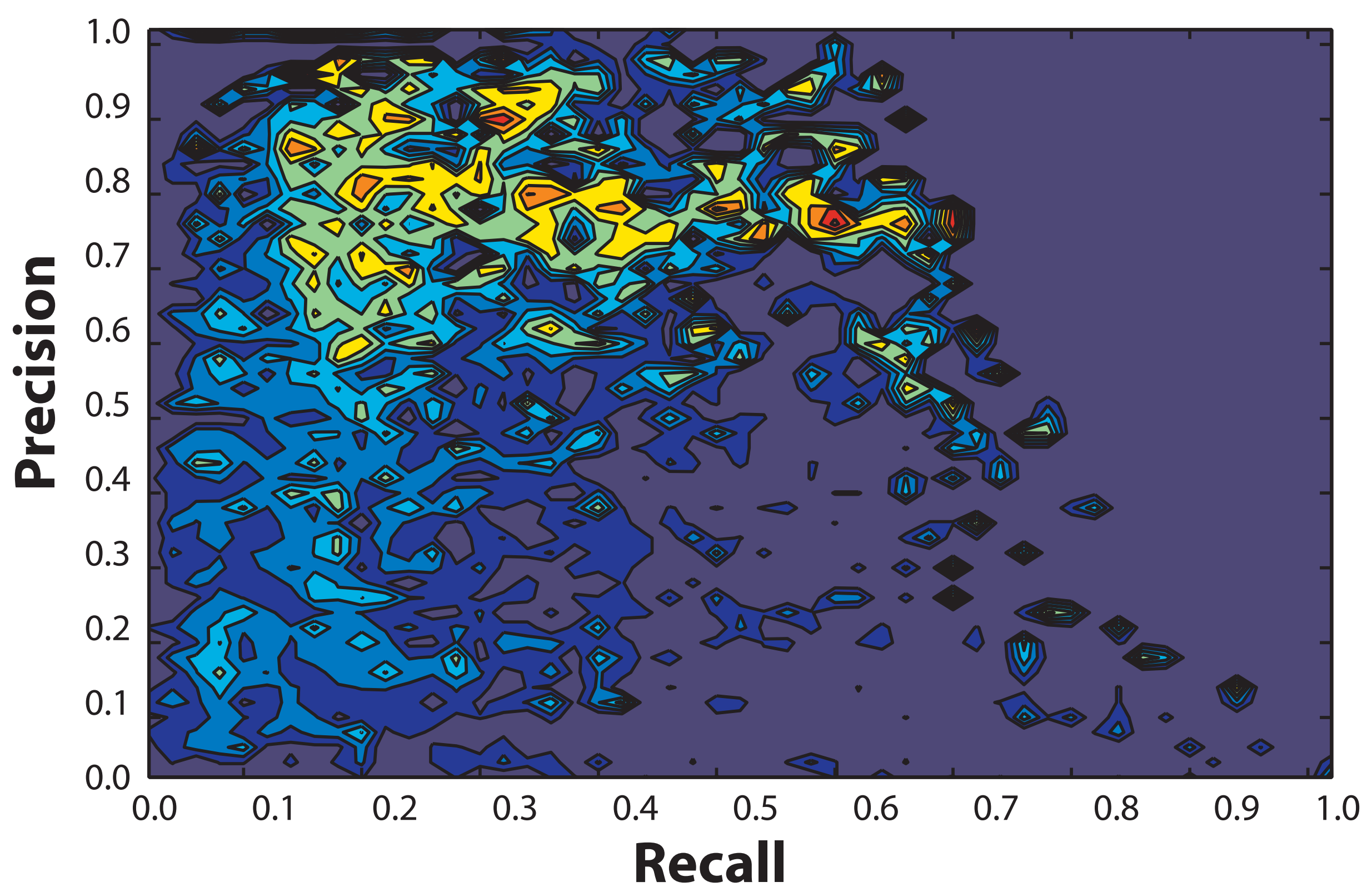
Lexical Properties	`phrase_contains_alphabetic_word(ab9121474_sen6_ph0).`
	`` phrase_contains_specific_word(ab9121474_sen6_ph1, `is'). ``
	`phrase_contains_originally_leading_cap(ab9121474_sen6_ph0).`

Word Frequency	`phrase_contains_some_arg_5x_word(ab9121474_sen6_ph1).`
	`phrase_contains_some_arg_2x_word(ab9121474_sen6_ph3).`

We use Aleph, an Inductive Logic Programming algorithm, to learn logical clauses that state that a relationship exists between two phrases if they are related by other background knowledge. Using the first positive example above as a starting point (or seed example), we can learn the following clause:

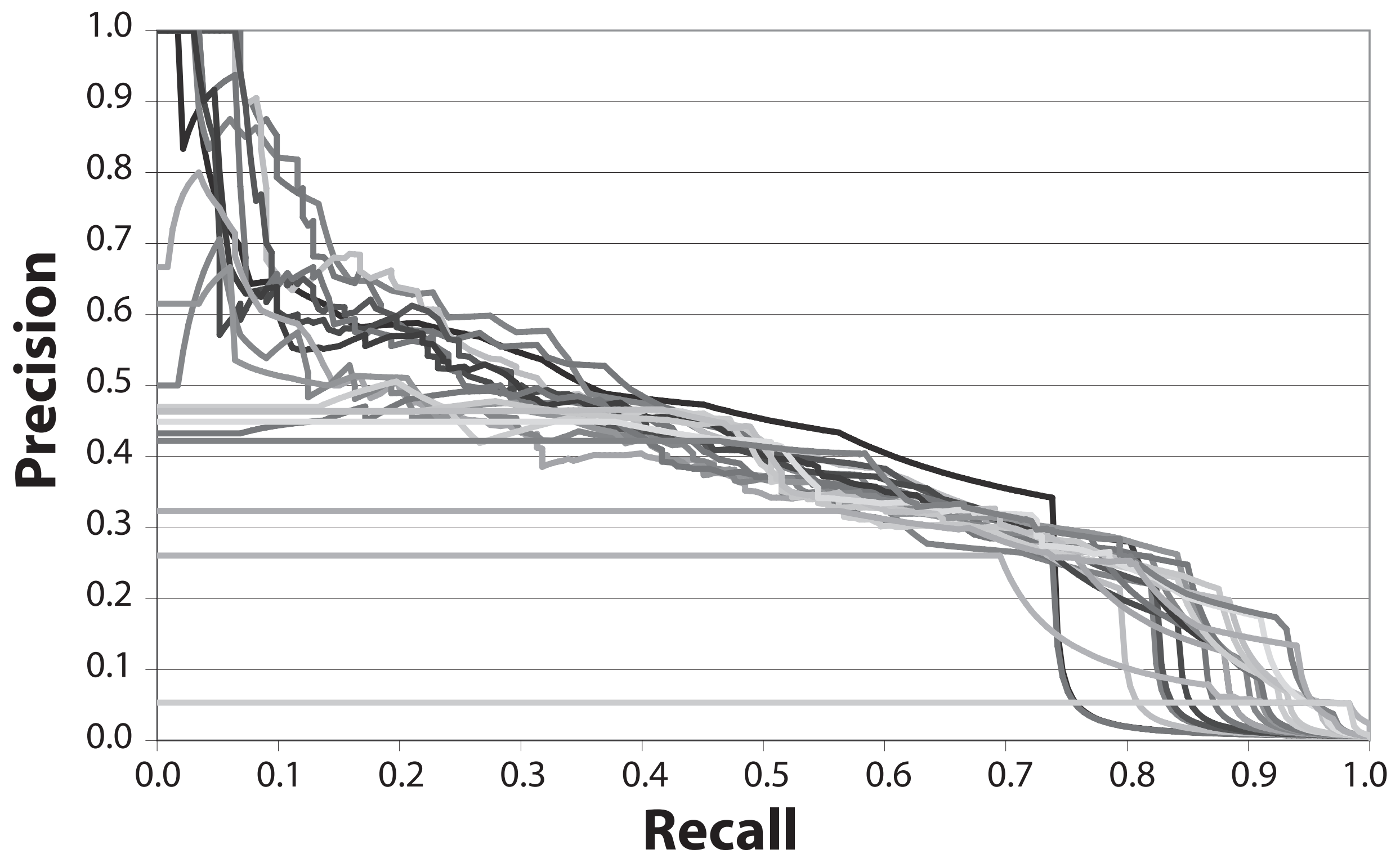
```
protein_location(P,L,S) :-
    target_arg1_before_target_arg2(P,L,S), first_word_in_phrase(L,A),
    phrase_contains_some_art(L,A), phrase_contains_some_marked_up_location(L,_),
    phrase_after(L,_), few_alphanumeric_words_in_phrase(P),
    few_alphanumeric_words_in_sentence(S), after_both_target_phrases(S,_).
```

Where Clauses Are Learned Using Gleaner



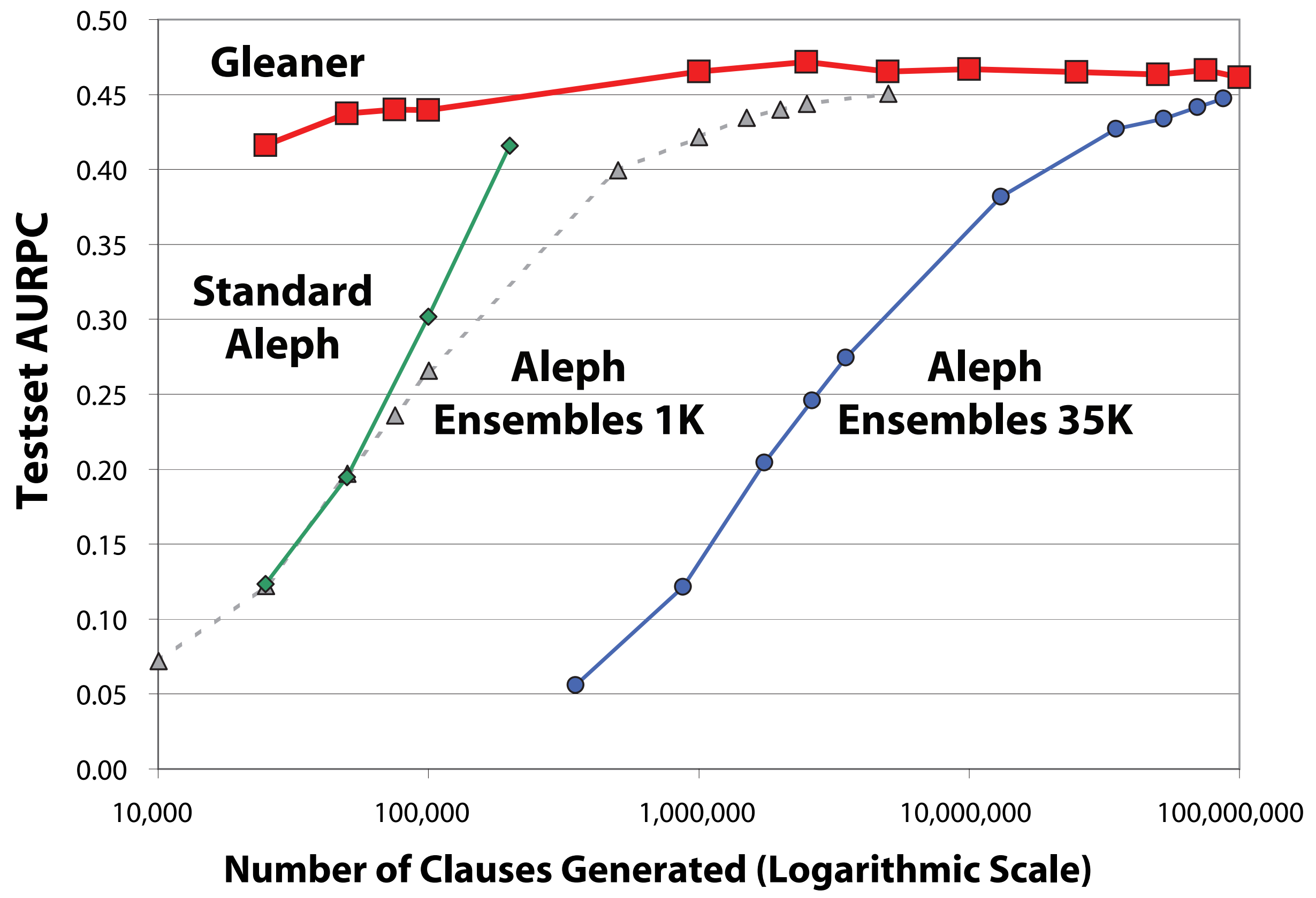
In fact, many unique clauses can be learned from one seed example. With such a large space of clauses, we use randomized search methods such as Rapid Random Restart to direct our search. Normally Aleph searches for the "best" clause given the seed example, one that covers as many positives and as few negatives as possible. Our algorithm Gleaner instead records the recall and precision of each visited clause, where recall is the percent of total positive examples covered, and precision is the percent of covered examples that are positive. Gleaner saves many clauses per seed, namely those that achieve the highest precision within a given recall range.

Ensemble Combination of Bin Theories



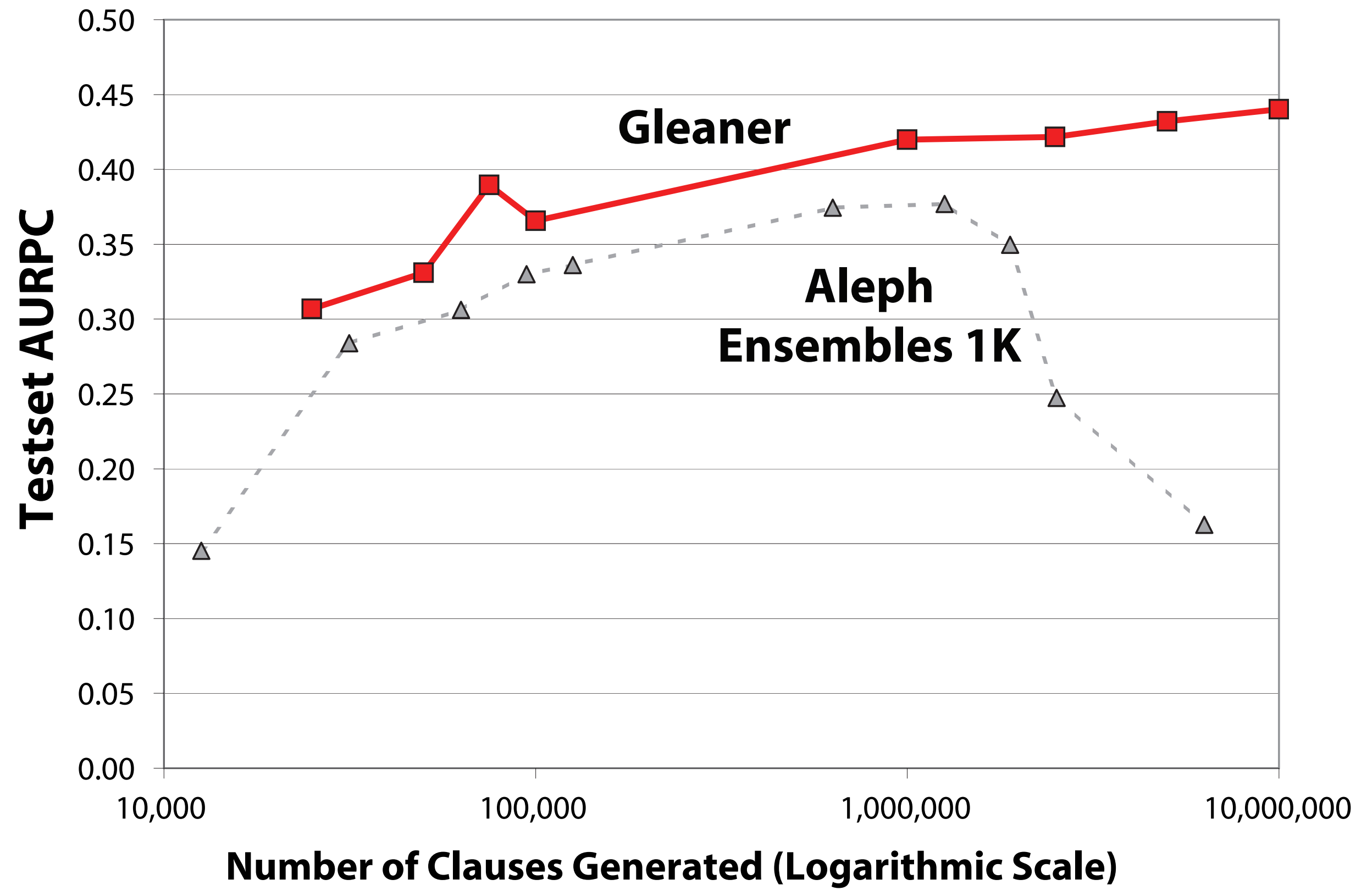
We collect clauses from 100 seeds in parallel, and then combine the learned clauses across all seeds into separate theories based on each clause's recall score. These theories are each evaluated on a tuning set as shown above to create many overlapping recall-precision curves. We build the final ensemble for the testing set by taking the theory with the highest precision for each recall range. Since each theory focuses on a different portion of the recall-precision space because of their internal clause composition, it is through the combination of these theories that Gleaner achieves an advantage in performance and time.

Yeast Protein Localization Dataset



We show here results from Gleaner on two Information Extraction datasets, the first on the localization of yeast proteins, and the second on the relationship between genes and diseases. We compare to standard ILP theories learned in Aleph as well as two variations of an ensemble method combining standard Aleph theories, using the Area Under the Recall-Precision Curve for each algorithm. Gleaner is shown to achieve higher AURPC scores than either comparison algorithm, often using an order of magnitude less clauses for equivalent performance.

Human Genetic Disorder Dataset



Our current work is focused on three main extensions. First, we are learning more diverse sets of clauses while maintaining the fast parallel nature of Gleaner in hopes of improving overall performance. Second, we are exploring alternate combination schemes for the learned clauses in our theories and giving weight to the clause's individual performance. Finally, we are combining the output of Gleaner with a Support Vector Machine framework to predict probabilities for each testing example, as opposed to our usual final AURPC metric.