Research Summary

Russell Greiner

Essentially all real-world software systems — ranging from programs that interpret visual information, to web-crawlers/data-miners that hunt for relevant information on the WWW, to expert systems that attempt to isolate and repair faults, and including of course autonomous agents performing such tasks — are expected to solve a sequence of tasks, each based on input from users and other external sources. A good system/agent is one that works effectively over the *distribution* of tasks encountered — *e.g.*, an interpretation system (resp., web-crawler, expert system) is good if it *usually* returns the appropriate labeling *for each image encountered* (resp., the most relevant articles for each request received, the most accurate diagnosis for each set of symptoms presented). Given that no system can work perfectly for all tasks, it is very useful to know this distribution when building the system, to construct the system best tuned to this set. Unfortunately, this distribution information is seldom known *a priori*.

In these situations, one can use *learning* techniques, which acquire the required information by observing the world, to build effective performance systems. Such learning techniques may, for example, build an effective expert system by observing the set of problems that will be posed to an expert system, and then incorporating (into the knowledge base) the information required to solve these problems. Similarly, a learning algorithm could produce a good interpretation system by assembling the vision modules found to be most effective at addressing the observed set of interpretation tasks; etc.

Unfortunately, most standard learning algorithms are rather limited and fragile. Many of my results extend these algorithms, and analyses, to produce more robust and more effective learning systems. In the last few years, I have built and analyzed many learning algorithms capable of handling realistic situations, including learners that can:

1. Find an effective performance system within a **combinatorial space** of systems
2. Exploit a given **initial theory**
3. Learn classifiers that can classify **partially specified** instances
4. Learn optimal **active** classifiers
5. Effectively use **relevance information**
6. Make **very efficient use** of training samples

The rest of this section overviews these contributions. Note that essentially all of these results include both theoretical analyses and empirical confirmations, obtained by applying implementations of the theories to real-world tasks.

1. Find an effective performance system within a *Combinatorial Space* of systems

As suggested above, we often measure the quality of a performance system/agent by how well it performs on average — *e.g.*, how often an expert system returns the appropriate diagnosis, or a web crawler finds the most relevant articles, etc. A learner's task is to find the best such agent, often from a large space of possible agents (*e.g.*, the learner may be seeking the best parameter setting, or the most appropriate set of heuristics to use, etc.). As it is often

difficult, or intractable, to find the globally optimal agent, many practical learning systems instead hill-climb to a local optimum. Even this task is problematic, as the hill-climber must know the distribution of tasks that will be encountered to decide whether to climb from one agent to another; unfortunately, this information is typically not known *a priori*.

The paper **[9]**[1] (which extends the AIJ-Award winning **[6]**) presents the PALO algorithm, which approximates this hill-climbing search when the "utility function" (used to evaluate each agent's performance) can only be estimated by sampling — and proves that PALO can efficiently return an agent that is, with provably high probability, essentially a local optimum. It also demonstrates the generality of this algorithm by sketching three meaningful applications, which provide concrete solutions to the utility problem from explanation-based learning, the multiple extension problem from non-monotonic reasoning and the tractability/completeness tradeoff problem from knowledge representation.

The subsequent papers **[14, 15]** show that a robot can use this same general idea, and algorithm, on the very different task of learning the best set of landmarks to use for registering its location. These papers also provide a large corpus of experiments that demonstrate that PALO works very effectively in this context as well.

2. Exploit a given *Initial Theory*

Most learning and data-mining algorithms build new classifiers "from scratch". This is clearly inefficient if one already has a good, but not completely correct, theory; here, a more efficient learner would instead begin with that initial theory, and revise it as required to accommodate new, more trusted information. This is the essence of the Machine Learning area of "Theory Revision".

In **[20]**, we describe the (now deployed) DELTA theory revision system, and show empirically that DELTA can effectively revise practical fielded theories, in realistic situations — *e.g.*, even when most training instances are missing many attribute values.

That system works by hill-climbing, rather than by directly seeking revisions whose accuracy is *globally* optimal. We chose this approach after proving that this task, of finding the globally optimal revision, is not just intractable, but is not even *approximatable* (*i.e.*, assuming $P \neq NP$, no efficient algorithm can find a revision that is even close to optimal) **[8, 10, 7]**.

3. Learn classifiers that can classify *Partially Specified* instances

Most theoretical analyses assume that both training and performance examples are complete — *i.e.*, that the value of every attribute is known to both learner and classifier. As noted above (and elsewhere throughout the learning and data-mining communities), real-world data is usually *incomplete*. The papers **[27, 23]** address this discrepancy by formally analyzing the task of learning to classify incompletely specified performance examples — considering, for example, the questions

[1] Each **[x]** number below points into the bibliography at the end of this summary; each "`($GreinerFTP/xxx.ps)`" pointer there expands to the URL "`ftp://scr.siemens.com/pub/learning/Papers/greiner/xxx.ps`".

Q: If the desired classification algorithm must classify partially-specified instances, which learning algorithm should be used?

Q: Should this learning algorithm use partially-specified instances (exactly like the ones its classifier will have to classify), or instances that have been "filled in" (*i.e.*, which have no missing values)? To be more concrete: will an intern learn more by (1) following a senior physician and seeing all-and-only what he sees, or (2) by reading a textbook, which supplies complete information about each patient?

Our papers show that the answers to both questions depend critically on *why* the attributes were missing: by a relatively benign process that simply flips coins to decide whether to block an attribute's value, or by a process that may base this decision on *e.g.*, the attribute's value. (*E.g.*, "bald men wear hats".) We also analyze the sample complexity of these situations, and provide empirical studies that validate our claims — *e.g.*, showing that "maximum likelihood estimation" works very well.

4. Learn optimal *Active* classifiers

Typical learners produce *passive* classifiers, which will simply return class labels, even if given partial information. By contrast, an *active* classifier can — at some cost — obtain the values of "blank" attributes. (*E.g.*, a data-miner seeking information may have the option of paying to use a restricted site.) Such an active classifier is evaluated based on both the cost spent acquiring information and the penalty paid for each mis-classification. In **[13]**, we consider the task of *learning* the best such active classifier, and present a few important situations where these classifiers can be learned efficiently. We then prove, however, that the general task is often *much* more difficult — often significantly more complex than learning the best passive classifier.

5. Use *Relevance Information*

Most analyses assume that missing values are *harmful*, as their omission can prevent both learner and classifier from seeing essential information. In some realistic situations, however, these omissions can be useful:

> Suppose a doctor uses a decision tree to diagnose patients, and records only the results from the tests actually run, along with the diagnosis reached — leaving the other attribute values blank. Afterwards, the learner's task is to reconstruct that tree, using only these "sparsely filled" records.

Here, the omitted values are omitted because their values are *irrelevant* to the classification. The papers **[22, 11, 12]** prove that that this "(ir)relevance" information significantly *simplifies* the learning task, as it allows a learner to "probably approximately correct" PAC-learn arbitrary decision trees, or even DNF formulae — two well-studied classes not known to be PAC-learnable in the standard model!

We also show how to handle small amounts of "degradation" of this relevance information, as well as (perhaps large) corruption in both the attribute values and the class labels; and also provide efficient algorithms that can revise a given imperfect theory within this model. (These positive results are in sharp contrast to the negative ones given in **2** above.)

6. Make *Very Efficient Use* of training samples

While the PAC-learning theory is very elegant, its sample complexity bounds — which bound

the number of training samples required to learn effectively — are so weak that they are not used in practice. The papers **[26, 25]** present new *sequential* learning procedures (which observe training examples one-at-a-time, and decide autonomously whether to halt and return a hypothesis, or continue training) and prove that these algorithms require many fewer training samples, while maintaining the exact same distribution-free worst-case guarantees. In **[24]**, we empirically demonstrate that their actual sample sizes are often *orders of magnitude* less than required by standard analyses — and are small enough that practitioners can actually use these algorithms for real applications! These results are therefore very important in typical learning and data-mining contexts, where labeled training examples are either expensive, or simply unavailable.

Other Activities

Of course this summary mentions only some of my contributions. In particular, it does not discuss my results in analogy **[5]**, knowledge representation (satisficing strategies **[16]**, diagnosis **[3]**, common sense reasoning **[2, 4]**), signal processing **[21]**, or control theory **[1]**.[2]

Let me close this section by mentioning some of my other activities. First, this summary does not discuss any of the proprietary work I have done at Siemens Corporate Research, which (broadly speaking) involves raising interest, and funds, to build large, fieldable systems that address real-world problems, spanning from adaptive decision support systems and Bayesian networks to vision-based tracking systems; and includes several pending patents. Second, I have maintained a very active service record outside of Siemens. In the last three years, I organized and edited a volume of the influential CLNL series **[17]**, and organized a very successful symposium on the topic of Relevance **[18]**, which included contributions from many prominent researchers representing Knowledge Representation, Machine Learning, Learnability, Information Retrieval, Statistics, Operations Research, and other fields. We are now producing a special issue of the *Artificial Intelligence* journal on this theme **[19]**. Over these years, I have also served, or am serving,

- as the Program Chair for the *International Symposium on Artificial Intelligence and Mathematics #5*
- as Workshop Chair and Tutorial Chair for both the *11th Machine Learning Conference* and *7th Computational Learning Theory Conference*,[3]
- on the thesis committee of 4 PhD and 1 MSc students in 5 universities (UofToronto, CMU, UofPennsylvania, Rutgers, UofWaterloo)
- on the tenure committee of 2 researchers
- on 7 conference committees
- on 3 workshop/symposium committees

in addition to the usual reviewing for (over a dozen) journals.

[2] Here, I mention only the relevant *journal* articles; my CV lists a number of other papers in prominent conferences, on these as well as other topics that range from solution caching, Horn approximations and the utility problem to plan verification and database theory.

[3] In addition to organizing and co-ordinating these events, I also obtained over $6000 in funding — enough to allow the entire community to attend for free.

Summary

While my results are quite broad and diverse, there is a clear unifying theme: using learning techniques to find practical solutions to realistic situations, often by exploiting (perhaps very subtle) information. As shown above, these contributions are applicable to many areas, including the exciting recent fields of autonomous agents and data-mining.

References

[1] P.E. Caines, R. Greiner, and S. Wang. Classical and logic-based dynamic observers. *IMA Journal on Control and Information*, 8:45–80, 1991.

[2] Charles Elkan and Russell Greiner. Book review of '*Building large knowledge-based systems: Representation and inference in the Cyc project*'. *Artificial Intelligence*, 61:41–52, 1993. (`$GreinerFTP/cyc.ps`).

[3] R. Greiner, B.A. Smith, and R.W. Wilkerson. A correction to the algorithm in Reiter's Theory of Diagnosis. *Artificial Intelligence*, 41(1):79–88, November 1989. Reprinted in "Readings in Model-based Diagnosis" (`$GreinerFTP/correct.ps`).

[4] Russell Greiner. Against the unjustified use of probabilities: A critique of Cheeseman's 'An inquiry into computer understanding'. *Computational Intelligence: An International Journal*, 4(1):79–83, February 1988.

[5] Russell Greiner. Learning by understanding analogies. *Artificial Intelligence*, 35(1):81–125, May 1988.

[6] Russell Greiner. Probabilistic hill-climbing: Theory and applications. In *Proceedings of the Ninth Canadian Conference on Artificial Intelligence*, pages 60–67, Vancouver, June 1992. Morgan Kaufmann. Awarded "AIJ Best Paper Prize". (`$GreinerFTP/phc-applic.ps`).

[7] Russell Greiner. The challenge of revising impure theories. In *Proceedings of the Twelfth International Machine Learning Conference*, 1995. (`$GreinerFTP/impure.ps`).

[8] Russell Greiner. The complexity of theory revision. In *Proceedings of IJCAI-95*, 1995. (`$GreinerFTP/comp-tr.ps`).

[9] Russell Greiner. PALO: A probabilistic hill-climbing algorithm. *Artificial Intelligence*, 83(1–2), July 1996. (`$GreinerFTP/palo_aij.ps`).

[10] Russell Greiner. The complexity of theory revision. *Artificial Intelligence*, 1997. Accepted subject to revision. (`$GreinerFTP/comp-tr-AIJ.ps`).

[11] Russell Greiner, Adam Grove, and Alex Kogan. Exploiting the omission of irrelevant data. In *Proceedings of the Thirteenth International Machine Learning Conference*, Bari, Italy, July 1996. Morgan Kaufmann. (`$GreinerFTP/superfluous.ps`).

[12] Russell Greiner, Adam Grove, and Alex Kogan. Exploiting the omission of irrelevant data. *Artificial Intelligence*, 1997. Accepted subject to revision. (`$GreinerFTP/superfluous-journal.ps`).

[13] Russell Greiner, Adam Grove, and Dan Roth. Learning active classifiers. In *Proceedings of the Thirteenth International Machine Learning Conference*, Bari, Italy, 1996. Morgan Kaufmann. (`$GreinerFTP/active-class-imlc96.ps`).

[14] Russell Greiner and Ramana Isukapalli. Learning to select useful landmarks. In *Proceedings of AAAI-94*, 1994. (`$GreinerFTP/useful-lms-aaai.ps`).

[15] Russell Greiner and Ramana Isukapalli. Learning to select useful landmarks. *IEEE Transactions on Systems, Man and Cybernetics, Part B*, 26(3), June 1996. (`$GreinerFTP/useful-lms-smc.ps`).

[16] Russell Greiner and Pekka Orponen. Probably approximately optimal satisficing strategies. *Artificial Intelligence*, 82(1–2):21–44, April 1996. (`$GreinerFTP/pao.ps`).

[17] Russell Greiner, Thomas Petsche, and Stephen J. Hanson. *Computational Learning Theory and Natural Learning Systems IV: Making Learning Systems Practical.* MIT Press, 1997.

[18] Russell Greiner and Devika Subramanian. Proceedings of the "Relevance" symposium. Technical Report FS-94-02, AAAI, Menlo Park, 1995.

[19] Russell Greiner, Devika Subramanian, and Judea Pearl. *Special AIJ Issue on "Relevance".* Elsevier, 1997. To appear.

[20] Pat Langley, George Drastal, R. Bharat Rao, and Russell Greiner. Theory revision in fault hierarchies. In *Proceedings of The Fifth International Workshop on Principles of Diagnosis (DX-94)*, New Paltz, NY, 1994. (`$GreinerFTP/th-rev.ps`).

[21] R. Lee, E. Milios, R. Greiner, J. Rossiter, and A.Venetsanopoulos. On the machine analysis of radar signals for ice profiling. *Journal of Signal Processing*, 18:371–386, 1989.

[22] R. Bharat Rao, Russell Greiner, and Thomas Hancock. Exploiting the absence of irrelevant information. In *AAAI Fall Symposium on 'Relevance'*, New Orleans, 1994. (`$GreinerFTP/superfluous.ps`).

[23] Dale Schuurmans and Russell Greiner. Learning default concepts. In *Proceedings of the Tenth Canadian Conference on Artificial Intelligence*, pages 519–523, 1994. (`$GreinerFTP/default.ps`).

[24] Dale Schuurmans and Russell Greiner. Practical PAC learning. In *Proceedings of IJCAI-95*, 1995. (`$GreinerFTP/pracpac-ijcai95.ps`).

[25] Dale Schuurmans and Russell Greiner. Sequential PAC learning. In *Proceedigs of COLT-95*, Santa Cruz, 1995. (`$GreinerFTP/seqpac-colt95.ps`).

[26] Dale Schuurmans and Russell Greiner. *Fast Distribution-Specific Learning*, chapter 10. MIT Press, 1997. (`$GreinerFTP/fast-disp-spec.ps`).

[27] Dale Schuurmans and Russell Greiner. *Learning to Classify Incomplete Examples*, chapter 6. MIT Press, 1997. (`$GreinerFTP/missing.ps`).