MACHINE LEARNING REDUCTIONS

JOHN LANGFORD

1. Some Motivation

In this section, we motivate the use of reductions in learning.

1.1. **Easy Learning.**

Charlie and Susan[1] each have a problem to solve: given information from house sensors, predict whether a user wants the light switched on or off. The problem has the following characteristics.

(1) Dependent data. Sensor values are correlated in time.
(2) Very rare events. Most of the time, light switch events don't occur, so natural data will have only a small set of significant events.
(3) Varying value events. Having the light unnecessarily switched on when you wake up at night is painful. Having the light switched on at dusk is usually fine.

Charlie solves the problem directly. He considers the problem carefully, constructing and tuning a model making predictions given the sensory inputs from a large number of examples. Charlie works for a year on this, and eventually gets a system that works reasonably well.

Susan solves the problem indirectly. She first notices that the problem is an instance of cost sensitive learning with dependent data. She applies a known reduction from cost sensitive learning to classification, tries a few classification algorithms, and picks the best. Susan spends only a day or two, producing a system that works reasonably well.

Susan's method (an automated machine learning approach) is much faster, requires much less training, takes advantage of past work by many researchers, and is easily verified. Charlie's method (a statistics-like approach) is surer and might be superior in the long run (or not. What if Charlie's sensors are obsolete by the time he succeeds?).

Reductions naturally aid Susan's method in several ways.

1.2. **Reductions are Reductionist.**

The fundamental reason why an investigation of reductions is merited is because it is inherently reductionist, and the reductionist approach to solving problems has often payed off very well in the long term. Examples of the reductionist approach include:

(1) Reducing computation to the transistor. All of our CPUs are built from transistors.
(2) Reducing rendering of images to rendering a triangle. Computers can now render near-realistic scenes in real time. The big breakthrough came from learning how to render triangles quickly.

[1] No Charlies or Susans were harmed in the making of this proposal.

(3) Many fields of science focus on theories making predictions about very simple systems. These predictions are then composed to make predictions about where space craft go, how large a cannonball needs to be, etc... Obviously, this approach has been quite successful.

1.3. **Why Now?** With all of this motivation, a basic question arises: Why hasn't this approach been more thoroughly investigated in the past? There are several answers.

(1) It has. There have been many attempts to use reductions to solve learning problems (detailed later in context). What is different about this proposal is using reductions *optimized for robustness*. In particular we want a robust transformation from subproblem predictions to problem predictions. Previous work on reductions has not emphasized this quality and the resulting performance is often spotty. In section 3.4 we show that this optimization often results in significantly better performance in practice.
(2) It's hard. A good understanding of what a learning problem *is* is difficult to acquire. This difficulty has frustrated many attempts to engage in reductions. The basic advantage we are working from is a relatively new mathematically based viewpoint on what the learning problem is.
(3) It's hopeless. In some sense reductions *can't* work for learning because there may always be some special domain knowledge not encoded into our learning algorithms. For this objection, it is important to remember that there are two distinct goals in machine learning: special purpose learning (where domain knowledge is helpful) and automated machine learning. The goal for reductions is automated machine learning: achieving good performance on as wide an array of learning problems as possible. There is hope that reductions can solve *many* of the common learning problems automatically, leaving hard problems for domain experts to confront without distractions.

With a few basic discoveries and a new understanding of reductions, we have the tools to attack this problem now.

2. The Proposal

The proposal is to produce a fully automatic learning system that can succeed as well as any other general approach on all supervised learning problems. There are several things to note about this goal:

(1) The goal has already been (more or less) met for binary classification problems. There are many easily automated binary classifiers that yield good performance in open competitions on classification problems.
(2) The goal is *not* "solve machine learning". As noted earlier, on any individual problem specialist knowledge about a problem can be invaluable in solving that particular problem. This form of machine learning can not be automated. A reasonable goal is creating a learning system that performs well on a broad array of learning problems (most of which are not naturally binary classification). Another reasonable goal is a learning system that performs well in competitions where specialist side information is not available, but humans can guide the learning process. This goal is more difficult, but may be feasible.

(3) This goal is one which inherently aims for a broad impact: simple usable code implies machine learning is available to nonspecialists. We believe that there are many more machine learning problems than machine learning specialists, and the key to unlocking these problems is a system such as this.

Three elements of this proposal remain to be specified.

(1) Specific subgoals. We discuss subgoals in the following subsections.
(2) Evidence of feasibility. Section 3 discusses several pieces of evidence that this is a feasible goal (in general and for us in particular).
(3) A plan. Section 4 discusses our plan of action. (There are three unfunded collaborators: Naoki Abe, Alina Beygelzimer and Bianca Zadrozny.)

2.1. **Mathematical Investigations.** There are a number of basic mathematical questions which we do not yet know the answer to. Progress here is principally measured in terms of papers and mathematical results.

2.1.1. *What are good new reductions?* MERIT: Most mathematical work is focused here. What is a robust reduction from ranking to classification? What is a robust reduction from structured prediction problems to classification?

BROADER IMPACT: Answers to this question provide new algorithms applicable to many learning problems. They give us the hope of automating another corner of machine learning problems.

2.1.2. *How can we think about reductions?* MERIT: This question is a deep research question. Much of our preliminary work has gone into discovering new theorem styles allowing us to analyze reductions (both old and new) for robustness.

BROADER IMPACT: New theorem styles give us a new lens with which to analyze, understand, and optimize reductions. Often, this can motivate entirely new reductions.

2.1.3. *Why do reductions work?* MERIT: This question is related to the previous. Many people have an intuition that reductions can create artificially hard subproblems that no learning algorithm will solve. This does not appear to be the case for the natural reductions even though it is easy to create unnatural reductions with this drawback. As an example, the existence of an "inversion reduction" [19] which undoes the original reduction is a good candidate.

BROADER IMPACT: A better method for analyzing the distinction between "good" and "bad" reductions helps us judge and create better reductions.

2.1.4. *Which learning problems are related by reduction?* MERIT: Currently, we know that reinforcement learning (with an oracle), cost sensitive classification, importance weighted classification, outlier detection, density estimation, ranking, regression, can all be reduced to classification. It is not known whether "pure" reinforcement learning can be reduced to classification. Clustering, manifold extraction, and feature selection all appear related to prediction, although perhaps not by reduction. Is that true? Which problems are *not* mutually reducible?

BROADER IMPACT: Answers to this question are primarily useful for organizing (and consequently teaching) via an objectively-defined zoology of learning problems.

2.1.5. *What should we reduce to?* MERIT: This is a deep research question. All of our investigations so far have been aimed at reducing towards binary classification. However, reductions are revealing significant mathematical evidence that reducing to regression may be more natural.

BROADER IMPACT: Determining which primitive is "right" is the sort of decision which may ultimately make-or-break this approach, in practice.

2.2. **Empirical Investigations.** Theory without an empirical investigation is often sidetracked into irrelevancy, something we wish to avoid. The goal is therefore to test various reductions for empirical tractability, efficiency, and efficacy on as wide an array of problems as feasible. This includes participation in contests (some of which we have already done).

2.2.1. *What does the mathematics miss?* Mathematics always abstracts away details. Sometimes these details are important, and the way to discover this is empirically.

MERIT: This is about honing our intuition for what is necessary.

BROADER IMPACT: Once we understand that an essential detail is left out, we can take it into account mathematically.

2.2.2. *How well do reductions work?* We have completed several empirical tests suggesting that several reductions work quite well. However, many are not sufficiently thoroughly tested (and there is still a clear need to invent new reductions). Further tests, especially in competitions, are warranted.

MERIT: This is essential to either discovering a bad idea and killing it off or showing the method works well.

BROADER IMPACT: Any attempt to create a widely useful system can only succeed with this sort of confidence-building (and debugging) measure.

2.2.3. *What should we reduce to?* This question appears the same as in section 2.1.5, but it is not. Mathematics might tell us what is a natural primitive from a reductions viewpoint. Unfortunately, the nature of learning problems seems to prevent discovering how exactly that primitive learner should behave. This is a question which must be answered (at least partially) empirically.

MERIT: The same as for refining and developing any other learning algorithm.

BROADER IMPACT: The learning algorithm developed here will, via reductions, be very broadly useful.

2.3. **Teaching.** Teaching is almost entirely about broader impact. The methods we propose to use include:

(1) Traditional classroom teaching. This has already been done at the University of Chicago for some of the preliminary work discussed in the next section.
(2) Untraditional classroom teaching. The Chicago 2005 Machine Learning Summer School has a session on reductions. The summer school has an expected 100 participants drawn from many locations on many continents. (I am organizing it.)
(3) Tutorials. A tutorial on this subject is going to be given at IBM and IJCAI in the summer of 2005.

Teaching is an ongoing process, so the material will be further developed over time with material available off a website.

2.4. **Software.** Developing easily usable software is the key to this project having a very broad impact. We plan to do this and make the code publicly available for common use.

2.5. **Workshop.** In 2003, a workshop on machine learning reductions was held at TTI-Chicago [25]. In 2006, we would like to have another workshop (at some location not yet determined). A workshop gives us-the-researchers a chance to think collectively about this topic which can be invaluable to the research process.

3. Why hope for success?

This project is broad, going all the way from deep mathematics to writing software. The goal of this section is to show that the team of researchers working on this is capable of accomplishing the proposal. There are several elements required for capability which we cover in the next subsections.

3.1. **Sufficient Collaborators (and research time).** Collaborators on this project include Naoki Abe, Alina Beygelzimer, and Bianca Zadrozny, (all Research Staff Members at IBM TJ Watson). See their attached letters in the appendix. Additional collaborators include Tom Hayes (NSF Postdoctoral fellow at Berkeley) and Varsha Dani (a graduate student at University of Chicago).

3.2. **Sufficient Expertise.** As we show in section 3.4, this group of collaborators has successfully engaged in research from inventing new theorem statements (what does it mean to optimize a reduction for robustness?) to software. This is a very broad range of expertise which has benefited our initial investigation.

3.3. **Sufficient Computational Resources.** TTI-Chicago is currently building a small cluster. The budget for the first year includes adding a share to the cluster. This should satisfy our computational requirements using the rule-of-thumb that algorithm development requires about an order of magnitude more computation than deployment.

3.4. **Sufficient hope the proposed method works.** This is the technical section. It is organized into subsections, one per reduction, from simplest to most complex. In these subsections, we show the theorem statement associated with each reduction and the empirical results (on standard learning datasets and in competitions). These subsections should be read in order, because they build on each other.

3.4.1. *Importance Weighted Classification to Binary Classification: Costing.* This section is about a trivial reduction which provides superior performance on the KDD 98 championship dataset. To define the reduction we must define what we reduce to and from.

Definition 3.1. (Binary Classification) Binary classification is defined by a measure D on the space $X \times \{0,1\}$ where X is an arbitrary feature space. The goal in classification is to find a classifier $c : X \to \{0,1\}$ which minimizes the error rate,

$$e(D,c) = \Pr_{(x,y)\sim D}(c(x) \neq y) \ .$$

Classification is a learning problem heavily studied both empirically and theoretically. For our purposes, it's important to note that this definition is completely general. In particular, any system producing a classification problem (x, y) can be described by some D (which we do not assume knowledge of). This D may, in fact, place all of its weight on a single example (x, y) and that example may even be chosen adversarially. It is typical to assume that examples drawn for training classifier c are drawn IID from the same distribution D as for testing. We specifically do *not* assume this here.

Definition 3.2. (Importance weighted classification) Importance weighted classification is defined by a measure D_i on the space $X \times \{0,1\} \times [0,\infty)$. The goal in importance weighted classification is to find a classifier $c : X \to Y$ which minimizes the expected importance of misclassified examples,

$$e_i(D_i, c) = E_{x,y,i \sim D_i} i I(c(x) \neq y) \ .$$

where $I()$ is the indicator function which is 1 when its argument is true and 0 otherwise.

Charlie and Susan's problem was importance weighted, and it is typical for the "real" classification problem to (in fact) be importance weighted classification. Note that this definition is similarly general and that when $i = 1$ for all samples, it is the same as classification. Despite the extreme generality of these definitions, we are able to relate them.

The ability to reduce importance weighted classification to classification is not too surprising (for example, it is assumed in [16]). The basic idea is to use the distribution $D(x, y, i) \propto iD_i(x, y, i)$. A good classifier according to D implies good importance weighted error with respect to D_i according to the following theorem.

Theorem 3.3. *(Importance Transform) For all classifiers c, for all importance weighted classification distributions D_i, let $D(x, y, i) \propto iD_i(x, y, i)$, then*

$$e_i(D_i, c) = e(D, c)E_{(x,y,i)\sim D_i}[i] \ .$$

This theorem is trivial in the sense of "well known folklore" and profound because it has a strong quantification "for all classifiers c, for all distributions D_i".

The import of this theorem is that if we can just change the distribution our examples are drawn from, importance weighted classification is binary classification. There are several methods for doing this, but it appears the "right" one for doing this is rejection sampling [27].

(1) Pick a constant c larger than any importance ($\forall i \ \ c > i$)
(2) For each sample (x, y, i), flip a coin with bias $\frac{i}{c}$. If the result is "heads" keep it, and otherwise discard it.

The advantage of this technique over resampling or duplication based approaches is that it preserves independence. In particular, if the samples of D_i were drawn IID, then the samples of the derived D produced by drawing from the D_i then rejection sampling are IID, which is not true of other approaches.

The algorithm "costing" [32] does rejection sampling t times, acquiring t sample sets drawn i.i.d. from D, learns a classifier on each data set, then predicts according to the majority vote of these classifiers. This simple algorithm achieves superior performance on the KDD-98 championship dataset, beating several algorithms hand-crafted to solve the problem, as shown in figure 1. The most surprising

Base Classifier	Profit
Naive Bayes	13163
Boosted N.B.	14714
C4.5	15016
SVM	13152

TABLE 1. Performance of the Costing reduction with various base classifiers on the KDD98 dataset. Default parameters were used for each base classifier. For historic reasons, the reported performance is in terms of "profit" on a test set (larger is better). For other details see [32].

thing about these empirical results is that a trivial algorithm (costing applied to a C4.5 which is a well known decade-old decision tree learning algorithm) achieves superior performance. The winner of the competition had a profit[2] of 14712. Other methods for solving this problem include algorithm alteration [14, 13] and methods based on Bayes risk theory [12, 23, 33]. To the best of our knowledge, no method has reported performance superior to the results observed here. This performance is particularly compelling because the computational cost of the Costing method is significantly lower than simply training on this dataset once and ignoring the importances. The reduction in training set size more than makes up for the t invocations of the oracle learning algorithm.

3.4.2. *Probabilistic Classification to Binary Classification: Probing.* This section is about a more complex reduction from probabilistic classification to importance weighted classification (see [20]). It can be composed with the Costing reduction to reduce probabilistic classification all the way to binary classification.

One technicality arises when working with this reduction: error rates are relatively meaningless. The problem is that the reduction creates classification problems which are inherently difficult due to large noise. In order to cope with this difficulty, we must work with regret instead.

Definition 3.4. (Binary Regret) The regret of a binary classifier c on an importance weighted distribution D_i is defined by:

$$r_i(D, c) = e_i(D_i, c) - \min_{c'} e_i(D_i, c')$$

Note that the definition of binary regret is again general. The definition of probabilistic classification is with respect to squared error.

Definition 3.5. (Probabilistic Classification) Probabilistic classification is defined by a measure D on the space $X \times \{0, 1\}$. The goal in probabilistic classification is to find a probabilistic classifier $c_p : X \to [0, 1]$ which minimizes the probabilistic squared error,

$$e_p(D, c) = E_{x,y\sim D}(D(y|x) - c_p(x))^2.$$

Note that the squared error is not generally observable because we do not generally know the conditional probability $D(y|x)$. However, the observable quantity: $E_{x,y\sim D}(y - c_p(x))^2$ is a D-dependent constant offset from $e_p(D, c)$. Consequently, minimizing one is equivalent to minimizing the other.

[2]Profit is a linear transformation of the importance weighted loss.

At a high level, the reduction algorithm "probing" that we use takes the original D and varies the marginal class probability over the range $[0,1]$ (discretized). For each choice of marginal class probability, a importance weighted classification problem is created and a classifier is learned. The predictions of each of these classifiers are combined by summing and dividing by the number of discretizations to arrive at a probability estimate. For an exact specification of the algorithm see [20]. Each of the steps here turns out to be critical to the ability to state the following theorem.

Theorem 3.6. *(Probing Transform) For all sets of classifiers $\{c\}$, for all classification distributions D,*

$$E_{x,y\sim D}(D(y|x) - \mathit{Probing}_{\{c\}}(x))^2 \leq 2E_{c_i\sim\{c\}}[r(D_i, c_i)]$$

where D_i is D with a marginal distribution altered as above and c_i is the ith classifier. Here $c_i \sim \{c\}$ is a uniform draw from the set $\{c_i\}$.

Just as in the last section, this theorem is strongly quantified, holding for all $\{c\}$(and consequently all learning algorithms producing c) and all D implying that it is generally applicable. The theorem states that minimizing error rate (and hence regret) on each individual classifier is equivalent to minimizing probabilistic squared loss. This is striking because error rate is observable while squared loss is not.

It turns out that this algorithm works quite well in practice, with the most significant drawback being the required computational time. A set of experimental results over a diverse set of datasets and classification algorithms is shown in figure 3.1. Probing on various base classifiers is compared with Bagging [9], fitting a sigmoid to a margin [29] (otherwise known as "Platt Scaling"), and the native multiclass predictions of the classifiers. The datasets are drawn from the UCI KDD archive [18], the UCI Machine Learning archive [8], and the KDD2004 championship.

Over all methods tested, probing with a decision tree worked better than any other (bagging with a decision tree was a close second). Although the theorem says nothing about log loss or "area under the ROC curve", it turns out that the probability estimates also appear to be (marginally to significantly) superior to all other common approaches for these metrics as well (see [20] for details).

This method was the basis we used to compete in the KDD2004 championship. We placed 2nd of the 20 teams that competed in both categories (or 6th of 60ish overall in each individual category). This performance may have been the best of all teams using a (near) fully automated approach—the winners effectively used a human as a part of the learning algorithm. Although this is good performance for an automated approach, we believe better performance is achievable with more research and thought.

3.4.3. *Multiclass Classification to Binary Classification: PECOC.* This reduction is from multiclass classification to binary classification (fully documented here [22]).

Definition 3.7. (Multiclass Classification) Multiclass classification is defined by a measure D_m on the space $X \times \{1, ..., k\}$ where X is an arbitrary feature space. The goal in multiclass classification is to find a classifier $c_m : X \to \{1, ..., k\}$ which minimizes the error rate,

$$e(D_m, c_m) = \Pr_{(x,y)\sim D}(c_m(x) \neq y) \ .$$

This problem is significant even when we allow ourselves to alter a binary learning algorithm directly. For example, the multiclass generalization of a support vector

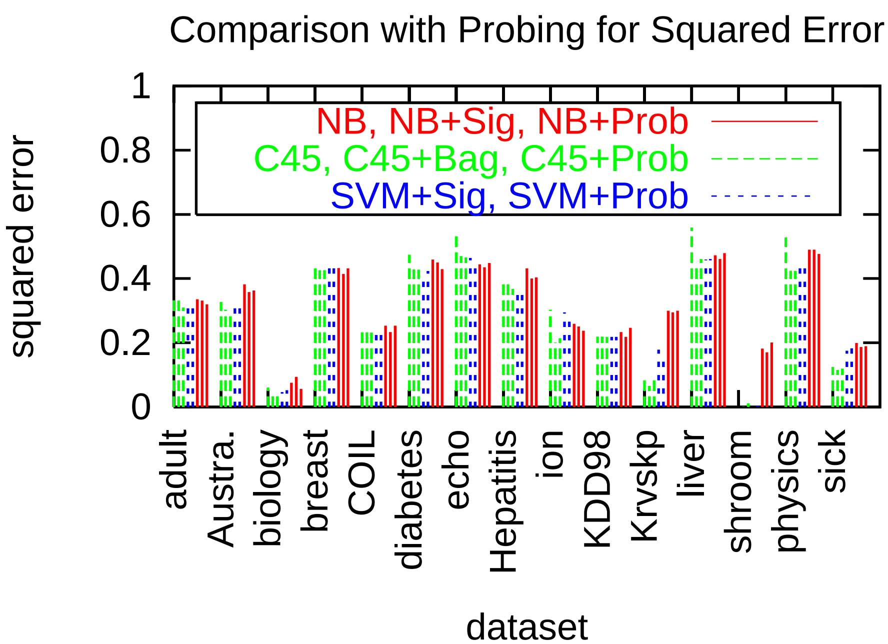

FIGURE 3.1. Experimental results for the probing algorithm (smaller is better). NB is Naive Bayes. C45 [30] is the C4.5 decision tree algorithm. SVM is the support vector machine (with a linear kernel). "Sig" is fitting a sigmoid to the margin. "Bag" is using Bagging to estimate probabilities. For each base classifier the bar plot is according to the order in the key.

machine was recently fixed [26]. Generic (black-box) techniques include Error Correcting Output Codes [11], the all-pairs approach, and the basic "one against all" approach.

Roughly speaking, ECOCs make multiclass predictions from a set of binary predictions of the form "is the label in set S or its complement?" where S varies with the binary prediction and the code. Any error correcting code can be used, although precisely which is best is not completely determined by the theory. The multiclass label which best agrees (according to hamming distance) with the binary predictions is chosen. It's easy to prove that there exist codes where average error rate ϵ implies multiclass error rate less than 4ϵ [2, 17].

One drawback of ECOC, is that in noisy situations optimal binary classifiers do not imply an optimal multiclass classifier via the ECOC procedure (see [22] for details). Theoretically, this difference occurs when analyzing the *regret*,

$$r(D_m, c_m) = e(D_m, c_m) - \min_{c'_m} e(D_m, c'_m)\ .$$

In particular, the counterexample theorem states that a small binary regret does not imply a small multiclass regret.

A simple fix for this is to use Probing to estimate the binary class probability distribution and then decode by finding the label with smallest l_1 distance to the

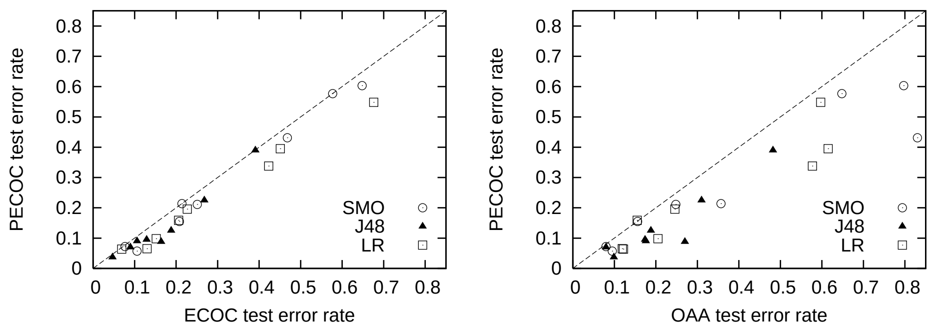

FIGURE 3.2. This is a plot comparing the multiclass test error rate of the PECOC reduction against ECOC reduction on Hadamard matrix code and the identity matrix code (which is typically called “one against all”). “SMO” is an SVM classifier, J48 is a decision tree classifier based on C4.5, and LR is Logistic regression. For all datasets and all binary learning algorithms, PECOC performance is superior.

predicted class probability. This can be composed with the probing reduction and costing to produce a reduction to binary classification. One valid concern is that this composition becomes very inefficient. However, since the l_1 distance can be regarded as an integration, the standard arsenal of tricks for approximate integration implies that the PECOC reduction is no more inefficient than the Probing reduction, in practice.

Theorem 3.8. *(PECOC Transform) For all sets of binary classifiers $\{c\}$, for all multiclass classification distributions D_m,*

$$r(D_m, PECOC_{\{c\}}) \leq 4\sqrt{E_{c\sim\{c\}}[r(D_c, c)]}$$

where D_c is the induced distribution on binary classifier c. Here $c \sim \{c\}$ is a uniform draw from the set $\{c\}$.

It turns out that this approach works quite well in practice as shown in figure 3.2. The test results are encouraging but not yet comprehensive—a comparison with other approaches for multiclass classification needs to be made.

3.4.4. *Cost sensitive classification to binary classification: SECOC.* This section is about a generalization of the PECOC reduction which handles the much broader set of all multiclass cost sensitive classification problems (documented here [22]). The theory of this reduction has been developed to the point of yielding an algorithm. Unfortunately, we do not yet have good experimental results, principally due to lack of datasets and freshness. The lack of datasets should not be interpreted as a lack of such problems in practice. Many recorded datasets have had useful cost-sensitive information dropped so that they could fit the paradigm of binary or multiclass classification. Previous work [12] (including some by the collaborators [6, **?**]) have addressed the lack of datasets by using artificially generated cost vectors, but this approach is not entirely convincing.

Definition 3.9. (Cost Sensitive Classification) Cost sensitive classification is defined by a measure D_s on the space $X \times [0, \infty)^k$ where X is an arbitrary feature space. The goal is to find a multiclass classifier $c_m : X \to \{1, ..., k\}$ which minimizes the expected cost,

$$e_s(D_s, c_m) = E_{x,\vec{s} \sim D} s_{c_m(x)} \ .$$

Naturally, we want to analyze the regret rather than the expected loss. The regret is defined as:

$$r_s(D_s, c_m) = e_s(D_s, c_m) - \min_{c'_m} e_s(D_s, c'_m)$$

The algorithm SECOC (Sensitive Error Correcting Output Code) [22], attempts to predict the expected sum of costs of a subset of the labels rather than the expected probability of membership in the set of labels. Decoding operates similar to PECOC, except that we choose the multiclass label minimizing the containing subsets expected sum of costs.

Theorem 3.10. *(SECOC Transform) For all sets of binary classifiers $\{c\}$, for all cost sensitive classification distributions D_s,*

$$r(D_s, SECOC_{\{c\}}) \leq 4\sqrt{E_{c \sim \{c\}}[r(D_c, c)] E_{(x,\vec{s}) \sim D_s} \sum_{i=1}^{k} s_i}$$

where D_c is the induced distribution on binary classifier c. Here $c \sim \{c\}$ is a uniform draw from the set $\{c\}$.

The SECOC transform results is similar to the PECOC transform, except that it is weakened by the expected sum of costs.

3.4.5. *The Kitchen Sink.* The previous subsections are only a distilled list of the reductions we have worked on. Others include:

(1) "Weighted All Pairs" [6] which reduces cost sensitive classification to binary classification in a loss-minimizing manner. Preliminary experiments show this approach works well in comparison to metacost [12].
(2) "Weighted One Against All" [7] which optimizes the one against all reduction from multiclass to binary classification, resulting in uniformly improved performance.
(3) A gradient-based reduction from cost sensitive classification to binary classification [1]. Preliminary experiments again show this approach works well in comparison to metacost [12].
(4) A "reduction" (in a weak sense) from reinforcement learning to cost sensitive classification [21]. This is mostly of theoretical appeal so far: it provides a more robust characterization of what a good policy is in terms of individual classification choices.

4. Research Plan

At a high level, the research plan is to investigate everything in the proposal (see section 2). This section provides extra information about prioritization, timing, and management. It is not meant to be a promise, because research is inherently opportunistic. Instead, it is the state of our current plans.

4.1. **Prioritization.** We have just completed analysis and preliminary implementation of a number of new algorithms. The priorities right now are essentially consolidating what we have discovered.

(1) Complete a tutorial on initial results to be presented at IBM and IJCAI in the summer of 2005.
(2) Release an initial public implementation of the software. We plan to use WEKA [31] as a source of classifiers.
(3) More thoroughly test various reductions to discover (empirically) best practices and refine software.

These steps should take only a few months (expected completion in November).

After this consolidation phase, we prioritize basic reductions work in the following order:

(1) Can reductions be used to solve complex structured learning problems? (one example is parsing)
(2) Can reductions be used to solve reinforcement learning problems?
(3) Can reductions be used to solve ranking problems? (for example, google has many inherently-ranking problems)

Naturally, we are also interested in solving the deeper mathematical questions around reductions. Work on these problems is driven by the difficulties encountered in solving the above problems.

As information (theoretical and empirical) is accumulated, we may shift the base of the reductions from classification to another primitive. This is not as dramatic an alteration as it might appear because all reductions theorems and software are modular.

4.2. **Timing.**

(1) May 2005-November 2005: Work on making reductions more available for common use. This includes software and experiments.
(2) Summer 2006: organize a workshop on reductions.
(3) November 2006: release a software library which can be called from any program as a subroutine.
(4) After November 2005:
    (a) Investigate new reductions algorithms and incorporate them into the software suite.
    (b) Investigate new primitive learning algorithms to better understand which can be used well with reductions.
    (c) Compete in various prediction competitions. As a sanity check on design, a well run competition is far superior to (possibly overfit) public datasets.

4.3. **Management: Who does what.** John Langford is the Priniciple Investigator for contact with NSF.

In the near term, all of {Alina, Bianca, John} are working on the tutorial for this project.

In the long term, John Langford will work on the following:

(1) Implementing and releasing the software suite (with feedback from others).
(2) Finding a graduate student (there is one possibility already) interested in working on this subject.

(3) Hiring an undergraduate at university of chicago for summer research. (There are several possibilities.)
(4) Investigate other loose ends.

Naoki Abe will investigate the following questions:

(1) Can reductions be used well for outlier detection?
(2) Can reductions be used well for density estimation?

Alina Beygelzimer will investigate the following questions:

(1) Is there a way to define reductions so it is impossible to create a reduction to an extra-hard problem?
(2) What is a good reduction from structured prediction to simpler learning problems?

Bianca Zadrozny will investigate the following questions:

(1) How can reductions be used to solve reinforcement learning problems?
(2) What tricks are necessary to make things work in practice which are not explained by the mathematics of reductions? How can these be formalized?

This split of activities is principally according to interest. Coordination is not a serious problem since everyone has worked together in the past, including working together remotely.

5. Results from Prior NSF Support

There is no prior NSF support for the PI or the unfunded collaborators.

6. Unfunded Collaborations

Naoki Abe, Alina Beygelzimer, and Bianca Zadrozny are unfunded collaborators who will be working on this project in the future. Their letters are included.

7. The Hopes for this Proposal

It is our hope that the propose work leads to many useful things.

(1) A new understanding of the relationship between different learning problems. This has obvious advantages with respect to teaching. Currently, many machine learning classes are of the 'zoo' variety where the instructor simply goes through topics which are difficult to organize in a rational way.
(2) New algorithms for solving learning problems. The reductions approach is particularly attractive for relatively esoteric learning problems because one reduction implies the availability of many algorithms. In addition, learning reductions are sufficiently basic so that most theory about the success or measurement of the oracle learning problem implies theory about the original learning problem.
(3) A newly lowered bar for ease-of-use in solving learning problems. Learning reductions provide a mechanism for solving a wide array of problems given that only problem definition knowledge. In other words, this is a mechanism which may provide machine learning to the masses.

It is also important to keep in mind what we do *not* hope for. We do not hope that reductions solve all learning problems. Any system accomplishing automated learning fails in comparison to systems incorporating prior information when that information is available. A simple extreme example of this is "predict the label from a semantically secure encryption of the label". As far as we know, this is generally not possible unless you know the secret key used to encrypt.

Learning reductions are mostly about extending learning (to a lower bar for use and to solving new problems) rather than replacing existing learning techniques. The mathematics of reductions is sufficiently basic that it generalizes any oracle learning algorithm to new learning problems. The oracle learning algorithm can use active learning, online learning, semisupervised learning, transductive learning, Bayesian learning, and other techniques. All of these styles of oracle learning algorithms can be used with all of the reductions discussed here.

The most fundamental reason why we are interested in this approach is given in section 1.2: reductions are reductionist. This approach to problem solving has paid off very well in many other fields, and it deserves investigation in learning.

References

[1] Naoki Abe, Bianca Zadrozny, and John Langford, "An Iterative Method for Multi-class Cost-sensitive Learning", KDD 2004.

[2] Erin Allwein, Robert Schapire, and Yoram Singer. Reducing multiclass to binary: A unifying approach for margin classifiers. *Journal of Machine Learning Research*, 1:113–141, 2000.

[3] Drew Bagnell, Sham Kakade, Andrew Ng, & Jeff Schneider, "Policy Search by Dynamic Programming", NIPS 2003.

[4] Peter Bartlett and Jon Baxter, "Estimation and Approximation Bounds for Gradient Based Reinforcement Learning", Technical Report, Australian National University, 2000.

[5] D.P. Bertsekas and J. N. Tsitsiklis "Neuro-Dynamic Programming", Athena Scientific, 1996.

[6] Alina Beygelzimer, Varsha Dani, Tom Hayes, John Langford and Bianca Zadrozny, "Weighed All Pairs", (in submission)

[7] Alina Beygelzimer, John Langford and Bianca Zadrozny, "Weighted One against all", AAAI 2005.

[8] C. L. Blake and C. J. Merz (1998). UCI Repository of machine learning databases. Irvine, CA: University of California, Department of Information and Computer Science.

[9] Leo Breiman, "Bagging Predictors", Machine Learning Journal, 1996, 24:2, 123-140.

[10] Hyeong Soo Chang, Robert Givan, and Edwin K. P. Chong, "On Line Scheduling via Sampling", AIPS 2000.

[11] T. G. Dietterich and G. Bakiri, "Solving Multiclass Learning Problems via Error-Correcting Codes", Journal of Artificial Intelligence Research, 2:263-286, 1995.

[12] Pedro Domingos, "MetaCost: A General Method for Making Classifiers Cost-Sensitive", KDD 1999.

[13] C. Drummon and R. Holte, "Exploiting the cost (in)sensitivity of decision tree splitting criteria", ICML 2000, 239-246.

[14] W. Fan, S. Stolfo, J. Zhang, and P. Chan, "Adacost: Misclassification Cost-Sensitive Boosting" ICML 1999, 97-105.

[15] Alan Fern, SungWook Yoon, and Robert Givan, Unpublished draft: http://min.ecn.purdue.edu/~afern/web-stuff/api-draft.ps

[16] Yoav Freund and Robert Schapire "A decision-theoretic generalization of on-line learning and an application to boosting", Journal of Computer and System Sciences, 55(1):119-139, 1997.

[17] Venkat Guruswami and Amit Sahai. Multiclass learning, boosting, and error-correcting codes. In COLT, 1999, 145–155.

[18] S. Hettich and S. D. Bay (1999). The UCI KDD Archive. Irvine, CA: University of California, Department of Information and Computer Science.

[19] Russel Impagliazzo, personal communications

[20] John Langford and Bianca Zadrozny, "Estimating Class Membership Probabilities Using Classifiers Learners", AISTAT 2005.

[21] John Langford and Bianca Zadrozny, "Relating Reinforcement Learning Performance to Classification Performance", in submission.

[22] John Langford and Alina Beygelzimer, "Sensitive Error Correcting Output Codes", COLT 2005.

[23] Dragos Margineantu, "Class probability Estimation and cost-sensitive classification decisions", ECML 2002, 270-281.

[24] Michail Lagoudakis and Ron Parr, "Reinforcement Learning as Classification: Leveraging Modern Classifiers" ICML 2003.

[25] Machine Learning Reductions Workshop, http://www.tti-c.org/workshop/machine_learning_reductions.html

[26] Yoonkyung Lee, Yi Lin, and Grace Wahba. Multicategory Support Vector Machines: Theory and Application to the Classification of Microarray Data and Satelite Radiance Data, Journal of the American Statistical Association, 99, 465 (2004) 67-81.

[27] John von Neumann, "Various Techniques Used in Connection with Random Digits", National Bureau of Standards, Applied Mathematics Series, 12, 36-38, 1951

[28] Andrew Ng and Jin Kim, Unpublished draft: http://www-2.cs.cmu.edu/~jcl/research/nips_RLtoCL/postnips.ps

[29] John Platt, "Probabilistic Outputs for Support Vector Machines and Comparison to Regularized Likelihood Methods", In A. Smola, P. Bartlett, B. Schölkopf, D. Schuurmans (eds .) *Advances in Large Margin Classifiers*, 61-74. MIT Press.

[30] J. R. Quinlan (1993). *C4.5: Programs for Machine Learning*, San Mateo CA: Morgan Kaufmann.

[31] I. H. Witten and E. Frank (1999). *Data mining: practical machine learning tools and techniques with Java implementations,* Morgan Kaufmann.

[32] Bianca Zadrozny, John Langford, and Naoki Abe, "Cost Sensitive Learning by Cost-Proportionate Example Weighting", ICDM 2003

[33] Bianca Zadrozny and Charles Elkan, "Learning and Making Decisions when Costs and Probabilities are Unknown", KDD 2001, 203-213.