PAC Analysis of Learning Weights in Multi-Objective Function by Pairwise Comparison

Ken Satoh
North 13, West 8 Sapporo 060-8628, Japan
Hokkaido University

abstract

This paper presents a theoretical analysis for a learning method of weights in multi-objective function. Although there are several learning methods proposed in the literature [Dyer79, Srinivasan73a, Srinivasan73b, Tamura85], none has yet been analyzed in terms of data complexity and computational complexity.
This paper steps toward this direction of giving a theoretical analysis on learning method for multiple objective functions in the viewpoint of the computational learning theory. As the first step, this paper presents a theoretical analysis of learning method of weights from pairwise comparisons of solutions[Srinivasan73a, Srinivasan73b]. In this setting, we show that we can efficiently learn a weight which has an error rate less than ϵ with a probability more than $1-\delta$ such that the size of pairs is polynomially bounded in the dimension, n for a solution, and ϵ^{-1} and δ^{-1}, and the running time is polynomially bounded in the size of pairs.

1 Introduction

In engineering domain, It is frequent that there are many objectives required to be optimal. For example, in making products, we have at least the following objectives:

1. Shortening a duration of making products.
2. Having lesser workers to make products.
3. Decreasing burden of workers.
4. Decreasing stocks of materials.
5. Making products as soon as requests come.

However, it is rare that an optimal solution is obtained in which every objective takes an optimal value; we often encounter situations where some of the objectives conflict each other. In the above example, to shorten a duration of making products and to have lesser workers, we might have to increase burden of workers, and to make products just in time, we might have to store some stocks of materials beforehand.

In such situations, all we can hope is to obtain a Pareto solution in which a value of an objective cannot be enhanced without sacrificing a value of another objective. However, there are usually many Pareto solutions and therefore choosing a solution is not a easy task. To solve this problem, we can sometimes use user's *preferences* over objectives. In other words, if there are conflicting objectives, then we choose a preferable solution based on these preferences.

This situation can be regarded as an optimization problem for combination of objectives and preferences. But, this optimization problem is different from the ordinal optimization problem in operations research since in the above situation, an object function is unknown. One approach for this problem is to learn multiple objective functions [Dyer79, Srinivasan73a, Srinivasan73b, Tamura85] where we decide the weights of combination of original objective functions to create "real" objective function.

It is very useful especially when we would like to transfer expert's preference into an expert system doing the above optimization problem in place of the expert. Unfortunately, however, the above researches only provide methods and evaluate empirically, so there are no theoretical analyses in the viewpoint of data complexity and computational complexity. Although their contributions are important, it is also important to know whether their approaches are computationally feasible or not, since a system using the above methods must learn in a permissible amount of data and time.

This paper steps toward this direction of giving a theoretical analysis on learning method for multiple objective functions in the viewpoint of the computational learning theory. As the first step, this paper presents a theoretical analysis of learning method of weights from pairwise comparisons of solutions[Srinivasan73a, Srinivasan73b]. The method has been applied to measurement of managerial success[Srinivasan73c] and preference of university administration [Hopkins77] and showed to be effective to some extent.

Our analysis for the method is an extension of the analysis in learning weights in similarity function in case-based reasoning[Satoh96] and learning preference relation in cardinality-based circumscription[Satoh00]. The analysis is based on PAC (probably approximately correct) learning [Valiant84]. In [Satoh96], we use relative distance information which tells if the distance between case A and case B is less than the distance between case A and case C and the algorithm

learns weights in a weighted Euclidean distance of the cases. In [Satoh00], we apply the above idea to learning preference relation for logical interpretations by regarding an logical interpretation as a case and a preference measure as a similarity measure between the interpretation and the most preferable interpretation. In this paper, we extend our previous results to learn weights of multi-objective functions of the more general form than those of [Satoh96] and [Satoh00].

2 Formal Analysis of the Learning Method

Let $A \in R^n$ be a solution and $u_i(A)(1 \leq i \leq t)$ be objective functions such that t is bounded by a polynomial of n and $u_i(A)$ are calculated in the time bounded by a polynomial of n. Let preference function $F(A, W)$ be of the form

$$F(A, W) = \sum_{i=1}^{m} W_i * F_i(u_1(A), ..., u_t(A))$$

where m is bounded by a polynomial of n, and W is a weight vector $(W_1, ..., W_m)$, and F_i is a polynomially evaluatable function of $u_1(A), ..., u_t(A)$. Note that since $u_1(A), ..., u_t(A)$ are calculated in the time bounded by a polynomial of n, each $F_i(u_1(A), ..., u_t(A))$ can be calculated in the time bounded by a polynomial of n. We assume that there exists a true weight vector $W^* = (W_1^*, ..., W_m^*)$.

Then, the learning problem is to find a hypothetical weight vector W which approximates W^* as possible. To do that, we provide our version of "approximation" of the true weight as follows.

Let $\mathbf{P}$ be any probability distribution over n-dimensional Euclidean space, R^n. Then, a set of different pairs between W and W^* is defined as follows:

$$diff(W, W^*) \stackrel{def}{=} \{\langle A, B\rangle \in R^n \times R^n | \\ (F(A, W) \geq F(B, W) \wedge F(A, W^*) < F(B, W^*)) \vee \\ (F(A, W) < F(B, W) \wedge F(A, W^*) \geq F(B, W^*))\}$$

The above set consists of solution pairs (A, B) such that (1) a solution A is actually preferable to the other solution B, but from the hypothesis weight, B is preferable to A or, (2) vice versa.

W is said to be an ϵ-approximation of W^* w.r.t. different pairs for $\mathbf{P}^2$, if the probability of $\mathbf{P}^2(diff(W, W^*))$ is at most ϵ. We call ϵ an *error rate*.

The following theorem shows that this framework is polynomially PAC-learnable.

Learn(ϵ,δ,m)
ϵ: accuracy, δ: confidence, m: the number of weights in the preference function
begin
Receive the definition of the preference function $F(A, W)$ with W unknown
and $max(\frac{4}{\epsilon}log_2\frac{2}{\delta}, \frac{8m}{\epsilon}log_2\frac{13}{\epsilon})$ pairs of solutions
and the results of comparison from the teacher.
for every pair (A, B)
if $A \leq B$ **then** add the following inequality to the constraint set:

$$F(A, W) \leq F(B, W)$$

if $B < A$ **then** add the following inequality to the constraint set:

$$F(B, W) + 1 \leq F(A, W)$$

Get consistent values for the above constraint set by linear programming
and output W.
end

Figure 1: Learning algorithm

Theorem 1 *There exists a learning algorithm which satisfies the following conditions for any probability distribution over R^n, $\mathbf{P}$, and an arbitrary constants ϵ and δ in the range $(0, 1)$:*

1. *The teacher selects a true weight vector W^* from $[0, \infty)^m$.*
2. *The teacher gives the definition of a preference function $F(A, W)$ with W unknown and gives N pairs according to $\mathbf{P}^2$ with the results of pairwise comparison defined by W^* to the algorithm.*
3. *The algorithm outputs a hypothetical weight vector W and the following hold.*
   - *The probability that W is not an ϵ-approximation of W^* w.r.t. different pairs for $\mathbf{P}^2$ is less than δ. We call δ a* confidence.
   - *The size of required pairs N for learning is bounded by a polynomial in n, ϵ^{-1} and δ^{-1}, and so is its running time.*

We show a learning algorithm of weights by binary comparison mentioned in the above theorem in Figure 1.

3 Experimental Results

We now show an experimental result under the following setting [1] .

1. $F(A, W)$ is defined as follows:

$$F(A, W) = \sum_{i=1}^{n} W_i * A_i.$$

   In other words, $t = 1$, $u_1(A) = A$, $m = n$, and $F_i(u_1(A)) = A_i$.
2. We use a randomized function to produce n values ranging over $(0, 1)$ and regard it as a true weight vector W^*.
3. We use a randomized function to produce n values ranging over $(0, 1)$ and regard it as a solution. We repeat this $2 * N$ times to produce N pairs of solutions.
4. Using the above algorithm, we learn a weight vector by using linear programming.
5. For the learned weight, we produce 10,000 test pairs randomly and calculate an error rate.
6. We repeat 100 times above and take the average of error rates.

We use UltraSPARC-IIi(440MHz) processor with 1GB memory for experiments.

Figure 2 shows relationship between the number of objective functions and the square root of learning time of weights. Since the graph is almost linear, the order of the learning time is $O(n^2)$ where n is the number of objective functions.

Figure 3 shows relationship between the number of objective functions and the error rate. The number of objective functions is almost proportional to the error rate.

Figure 4 shows relationship between the size of training pairs and the inverse of error rate. This graph is almost linear, so the size of training pairs is almost inversely proportional to the error rate.

Figure 5 indicates that the required total size of training pairs is almost $\frac{C * n}{\epsilon}$ on average in order to obtain the average error rate, ϵ, where C is a constant. From the graph, $C \approx 0.488$. This size for training pairs is smaller than the size in our PAC-learning analysis. It is probably because in PAC-learning analysis, we consider the worst case, while in the experiment, we assume that the probability distribution is fixed where the behavior may not be so pathological.

To summarize, the experiments show that the computational complexity of learning weights is

[1] This is another interpretation of the experiment showed in [Satoh00].

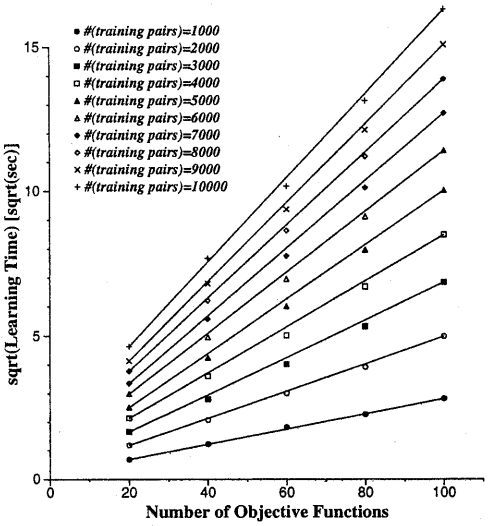

Figure 2: Relationship between n and $\sqrt{LearningTime}$

$O(n^2)$ and the data complexity is $O(n/\epsilon)$. We believe that the results are very encouraging.

4 Conclusion

This paper presents a computational analysis of a learning method for weights in multi-objective functions which uses pairwise comparisons. The analysis shows that the learning method can polynomially PAC-learn the weight. Therefore, we can say that the learning method is feasible in terms of the worst-case analysis in computational learning theory.

We pursue the following as future works.

1. Applying our method to real application domain.
2. Making our framework robust to errors on comparison.
3. Analyzing our method theoretically when the distribution is fixed to explain the experiment result in this paper.

Reference

[Blumer89] Blumer, A., Ehrenfeucht, A., Haussler, D., and Warmuth, M. K., "Learnability and the Vapnik-Chervonenkis Dimension", *JACM*, **36**, pp. 929 - 965 (1989).

[Dyer79] Dyer, J. S., and Sarin, R. K., "Measurable Multiattribute Value Functions", *Operations Research*, Vol. 27, No.4, pp. 810 - 822 (1979).

[Hopkins77] Hopkins, D. S. P., Larrenche, J. C., and Massy, W. F., "Constrained Optimization of a University Administrator's Preference Function",

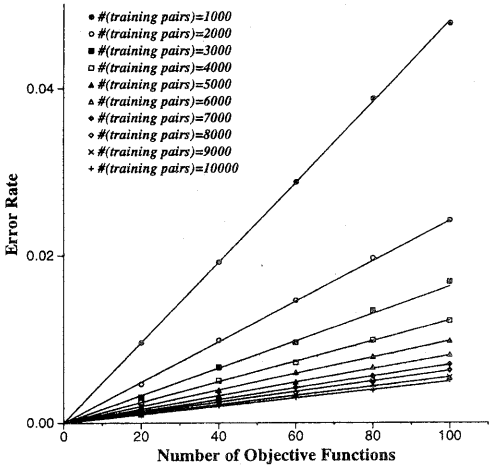

Figure 3: Relationship between n and ϵ

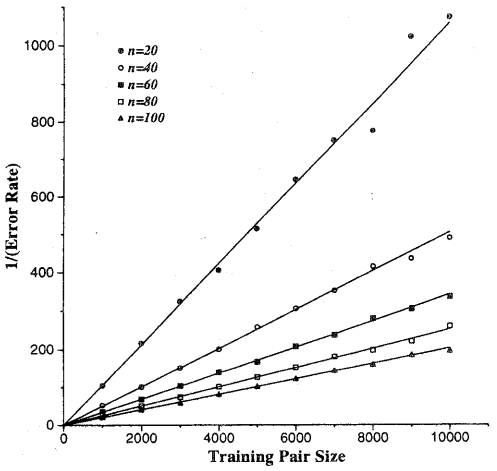

Figure 4: Relationship between number of training pairs and ϵ^{-1}

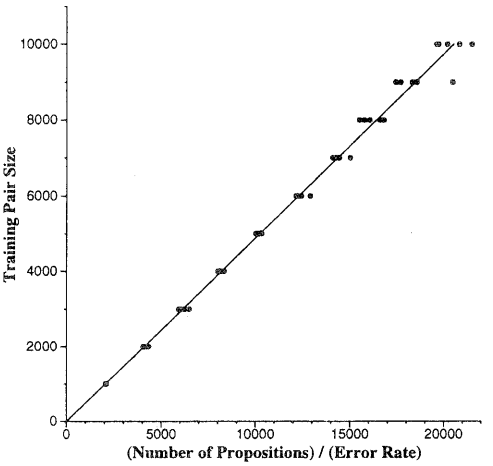

Figure 5: Relationship between n/ϵ and number of training pairs

Management Science, Vol. 23, No.11, pp. 1161 – 1168 (1977).

[Karmarker84] Karmarkar, N., "A New Polynomial-time Algorithm for Linear Programming", *Combinatorica*, **4**, pp. 373 – 395 (1984).

[Satoh96] Satoh, K., Okamoto, S., "Learning Weights in a Similarity Function from Distance Information", Journal of Japanese Society of Artificial Intelligence, Vol. 11, No.2, pp. 238 – 245 (1996) (in Japanese).

[Satoh00] Satoh, K., "PAC-learning of Preference Relation over Interpretations in Lazy Nonmonotonic Reasoning", H. Motoda and S. Muggleton (eds)., Machine Intelligence 15, Oxford University Press. (to appear) (2000).

[Srinivasan73a] Srinivasan, V. and Shocker, A., "Linear Programming Techniques for Multidimensional Analysis of Preferences", *Psychometrika*, Vol. 38, No.3, pp. 337–369 (1973).

[Srinivasan73b] Srinivasan, V. and Shocker, A., "Estimating the Weights for Multiple Attributes in a Composite Criterion using Pairwise Judgments", *Psychometrika*, Vol. 38, No.4, pp. 473–493 (1973).

[Srinivasan73c] Srinivasan, V., Shocker, A., and Weinstein, A. G., Measurement of a "Composite Criterion of Managerial Success", *Organizational Behavior and Human Performance*, Vol. 9, pp. 147 – 167 (1973).

[Tamura85] Tamura, H. and Hikita, S., "An Interactive Algorithm for Identifying Multiattribute Measurable Value Functions based on Finite-Order Independence of Structural Difference", Transactions of SICE (in Japanese), Vol 29, No. 11, pp. 62 – 68 (1985).

[Valiant84] Valiant, L. G., "A Theory of the Learnable", *CACM*, **27**, pp. 1134 – 1142 (1984).