A Flexible Classifier Design Framework based on Multi-Objective Programming

Sibel Yaman, Student Member, IEEE, and Chin-Hui Lee, Fellow, IEEE

Abstract—We propose a multi-objective programming (MOP) framework for finding compromise solutions that are satisfactory for each of multiple competing performance criteria in a pattern classification task. The fundamental idea for our formulation of classifier learning, which we refer to as iterative constrained optimization (ICO), evolves around improving one objective while allowing the rest to degrade. This is achieved by the optimization of individual objectives with proper constraints on the remaining competing objectives. The constraint bounds are adjusted based on the objective functions obtained in the most recent iteration. An aggregated utility function is used to evaluate the acceptability of local changes in competing criteria, i.e., changes from one iteration to the next. Although many MOP approaches developed so far are formal and extensible to large number of competing objectives, their capabilities are examined only with two or three objectives. This is mainly because practical problems become significantly harder to manage when the number of objectives gets larger. We, however, illustrate the proposed framework in the context of automatic language identification (LID) of 12 languages and 3 dialects. This LID task requires the simultaneous minimization of the false-acceptance and false-rejection rates for each of the 15 languages/dialects, and, hence, is an MOP problem with a total of 30 competing objectives. In our experiments, we observed that the ICO-trained classifiers result in not only reduced error rates but also a good balance among the many competing objectives when compared to those classifiers that minimize an overall objective. We interpret our experimental findings as evidence for ICO offering a greater degree of freedom for classifier design.

Index Terms—pattern recognition, applications of multi-objective programming, automatic language identification.

I. INTRODUCTION

It has been increasingly recognized that realistic problems often involve the consideration of a tradeoff among many conflicting goals. Traditional machine learning algorithms aim at satisfying multiple objectives by combining the objectives into a global cost function, which in most cases overlooks the underlying tradeoffs between the conflicting objectives. Such single-objective programming (SOP) approaches promise that the chosen overall objective function is optimized over the training samples. At first place, it is often not easy to combine all the competing criteria in a single overall objective function. Furthermore, there is no guarantee on the performance of the individual metrics for they are not considered separately. For these reasons, methods of traditional single objective optimization are not enough.

The multi-objective programming (MOP) offers new horizons for solving problems with competing objectives [1]. The mathematical foundations for MOP were already laid by Pareto more than a hundred years ago in Economics, and numerous methods have been developed [2], [3]. However, MOP algorithms were not extensively studied in machine learning until mid 1990s. Three criteria in designing neural networks for system identification were studied in [4] to obtain a proper balance between accuracy and model complexity. For similar purposes, the squared error and the norm of the weight vectors in a neural network were minimized simultaneously in [5]. Evolutionary MOP of support vector machines (SVMs) was considered in [6] to minimize false-acceptance rate, false-rejection rate and the number of support vectors to reduce model complexity. These approaches are illustrated with only two or three competing objectives, and what happens with more objectives remains to be explored. In [7], the authors developed a family of SVMs using goal programming (GP). GP is a branch of MOP where deviations of objectives from some pre-selected target levels are minimized. As a shortcoming, the selection of target levels for the objectives requires prior knowledge about the problem. In [8], the generation of neural networks based on the receiver operating characteristics (ROC) analysis was investigated using an evolutionary algorithm. Evolutionary algorithms are meta-heuristic optimization algorithms, which often lack mathematical analysis.

In this paper, we develop an analytical MOP framework, called Iterative Constrained Optimization (ICO), for finding best compromise solutions among as many as 30 competing performance criteria. Our approach is inspired by the following facts:

- We require each objective function to attain a satisfactory level.
- We want to have the flexibility to achieve different levels of tradeoff.
- It is hard to determine a realistic overall objective function a priori.
- One objective tends to dominate in an SOP problem even when an overall objective function is realistically determined.

MOP methods mainly fall into two major categories, where the original MOP problems are converted into SOP problems either by aggregating the objective functions into an overall objective function or by reformulating the problem with proposer constraints [1]. In ICO, each one of the objectives is iteratively optimized one after another with constraints on others, and the constraint bounds are adjusted by using the objective functions attained in the most recent iterate. It then becomes possible to tradeoff the performance in already-good objectives to improve the remaining not-so-good objectives. By doing so, a better balance among many competing objectives can be achieved.

S. Yaman and C.-H. Lee are with the School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA, 30332-0250 (email: syaman@ece.gatech.edu; chl@ece.gatech.edu).
where each objective is comparably good. On the contrary, SOP techniques may favor or sacrifice some of the objectives for the sake of improving the selected overall objective.

In real-life problems it is often beyond computation, if not impossible, to find out how small changes in some objectives affect others, and hence to determine whether resulting solution is more preferable than the starting solution. This is because when one objective worsens, some of the remaining objectives improve and some worsen. That is why, when the number of objectives increases beyond two, it becomes considerably more challenging to develop algorithms capable of finding good compromise solutions. It is for this reason that in the above-mentioned MOP approaches, the treatment is formal, and could be extended to more objectives, however, only two or three competing criteria are studied to illustrate the developed algorithm.

When the number of objectives increases, the major difficulty is the necessity of being able to control how much the individual objectives change in each update iteration. In this paper, we formulate a step-size control strategy motivated by the fundamental concepts from the utility theory [9]. We first illustrate the use of ICO with only two conflicting objectives, namely to minimize false rejection and false acceptance rates in automatic language identification (LID) task. We later demonstrate that ICO framework is able to generalize to handle as many as 30 competing objectives for LID, where the goal is to achieve minimal false-rejection and false-acceptance rates for 15 languages.

This paper is organized as follows: In Section II, we present the connection between the general classification problems and MOP. In Section III, we formulate the MOP problem and present some MOP terminology. In Section IV, we describe the details of ICO approach. In Section V, we illustrate the use of ICO problem in an LID task. We report our experimental results and discussions in Section VI. We finally present our concluding remarks in Section VII.

II. CLASSIFICATION PROBLEM WITH MULTIPLE COMPETING OBJECTIVES

We begin by considering a binary classifier, which is a mapping where each data sample \( x \in S \subseteq \mathbb{R}^n \) is labeled with one of two class labels, i.e., with either \( C_+ \) or \( C_- \). A classification error occurs when a sample from one class is labeled as that of the other class. The classification performance of such a binary classifier can be summarized by the false-acceptance (FA) and false-rejection (FR) rates. FA rate corresponds to the probability of labeling a negative sample as a positive sample, and FR rate corresponds to the probability of labeling a positive sample as a negative sample. The binary classification problem is, thus, an MOP problem with the FA and FR rates as two competing objectives. It necessitates a trade-off among these two objectives as any reduction in FR rate comes at the expense of an increase in FA rate, and vice versa.

When solving an \( M \)-class classification problem, one needs to cope with complications resulting from increased number of objectives. An example of an \( M \)-class classification problem is the automatic language identification (LID) task [10], [11]. LID is the problem of identifying or detecting the language being spoken by an unknown speaker from a sample of speech. Many applications such as directory assistance or automatic translation makes substantial use of identifying the language being spoken. When the LID task is detecting the language spoken in a speech utterance, the goal is reducing the FA and FR rates as much as possible for each of the target languages. This corresponds to the minimization of a total of \( 2M \) objectives, where each individual objective is either an FA or an FR rate for the target languages and dialects.

If one needs to roughly compare two LID systems, this can be done via comparing the average error rates, \( E^{avg} \), given by:

\[
E^{avg}(\Lambda) = \frac{1}{2M} \sum_{m=1}^{M} [E^{FR}_m(\Lambda) + E^{FA}_m(\Lambda)],
\]

where \( E^{FR}_m(\Lambda) \) and \( E^{FA}_m(\Lambda) \) are the FA and FR rates of the \( m^{th} \) language, respectively. In general, building classifiers which minimize such an overall performance criterion often fails to meet other design goals. For example, it is possible to obtain a low average error rate when one or some of the individual error rates are unacceptably high. This is mainly because some of the objectives dominate the optimization process. However, sacrificing any objective in exchange of an improvement in others is not tolerable in most real-life problems, since each objective is important for the successful operation of the overall system.

In this paper, we address the important issue of how to design classifiers by taking into account multiple quality objectives to be optimized. In designing such classifiers, we make use of the minimum classification error rate (MCE) framework, which has been extensively studied in automatic speech recognition field [12], [13]. Recently, MCE based training has been applied to several domains where different performance metrics are of interest. In [14], an approach named maximal figure of merit (MFoM), is described where classifiers are trained to optimize one of the standard performance metrics of text categorization, namely precision, recall or F1 measure. Linear classifiers are discriminatively trained using the MCE framework for the task of automatic image annotation in [15], and for the task of automatic language identification in [16]. In [17], discriminant functions derived from conditional random fields are trained with MCE method for text chunking and named entity recognition tasks. MCE framework develops a general framework so that any chosen single performance objective can be optimized. This is achieved by first approximating the given performance metric as differentiable functions of the classifier model parameters. Nevertheless, MCE is a formulation for the optimization of a single objective.

Before talking about the details of the developed framework, we find it convenient to present the notation used in our treatment in a look-up table as in Table I.

III. MOP TERMINOLOGY

Suppose that we are given a set of \( K \) competing objectives, \( f_k(\Lambda) \in (0, 1) \), \( k = 1, ..., K \), each of which is nonlinear
function of a set of $M$ decision vectors, $\omega_m \in \mathbb{R}^n$, $m = 1, ..., M$. The best compromise solutions, $\{\hat{\omega}_1, ..., \hat{\omega}_M\}$, are found by MOP, which is formulated as

$$\hat{\Lambda} = \{\hat{\omega}_1, ..., \hat{\omega}_M\} = \arg \min_{\Lambda} \left[ f_1(\Lambda), f_2(\Lambda), ..., f_K(\Lambda) \right]. \quad (2)$$

In general, an improvement with regard to one objective causes a deterioration of another. This corresponds to the situation that the objective functions are at least partially conflicting, meaning that they are conflicting at least in some regions of the search space. In this paper, we refer to such objective functions as competing objectives.

In single objective optimization problems, we say that a solution with a smaller objective function value is better than one with a large objective function value. However, in MOP problems, there is no natural ordering in the objective space. For example, let the vector $(f_1, f_2)$ denote the objective function values in an MOP problem with two competing objectives. The vector $(1, 1)$ can be said to be less than $(3, 3)$, but it is not obvious how to compare $(1, 3)$ and $(3, 1)$. In MOP problems, there are usually (infinitely) many optimal compromise solutions that form the so-called Pareto optimal set. A decision vector $\Lambda$ is (global) Pareto optimal if there does not exist another decision vector $\Lambda'$ such that $f_k(\Lambda') \leq f_k(\Lambda)$, for all $k = 1, ..., K$, and $f_p(\Lambda') < f_p(\Lambda)$ at least for one index $p$ [1]. According to the definition of Pareto optimality, moving from one Pareto optimal solution to another necessitates trading off. Mathematically, every Pareto optimal solution is an equally acceptable solution to the MOP problem.

MOP methods mainly fall into two major categories, where the original MOP problems are converted into SOP problems either by aggregating the objective functions into an overall objective function or by reformulating the problem with proposer constraints [1]. One of the most common engineering practices to solve MOP problems is the so-called weighting method. It reformulates the original MOP problem as a convex linear combination of the individual objectives with positive weights, $\gamma_k \geq 0$, such that $\sum_k \gamma_k = 1$. The task is, then, to minimize this overall objective function, i.e.,

$$\min_{\Lambda} \sum_{k=1}^{K} \gamma_k f_k(\Lambda), \quad (3)$$

where the weights, $\gamma_k$, reflect the significance of the individual objectives. One of the fundamental limitations of the weighting method is that the feasible objective space is not necessarily convex whereas the Pareto optimal solutions in the non-convex subset of Pareto optimal solutions cannot be found with the weighting method [18]. Another major drawback is that solving the problem with numerous weight vectors will give a limited number of Pareto optimal solutions. It is crucial that these solutions be spread in the objective space as uniformly as possible. The weighting method fails to meet this requirement and generates an irregular discretization of the (convex part of the) Pareto optimal set [1], [18].

IV. ITERATIVE CONstrained OPTimization (ICO)

Consider instead formulating the multi-objective optimization problem as a set of $K$ single-objective optimization problems in the form

$$\min_{\Lambda} f_k(\Lambda)$$

subject to $f_p(\Lambda) \leq \bar{f}_p$, $p = 1, ..., K, p \neq k$, \quad (4)

for all $k = 1, ..., K$, i.e., the minimization of one objective function, $f_k(\Lambda)$, with constraints on the other $(K - 1)$ competing objectives, $f_p(\Lambda)$, where $\bar{f}_p$’s are the upper bounds for the objective function values to be attained. The tightness of the constraints specify the size of the search region. On one hand, loose constraints may result in more degradation in the other objectives than desired. On the other hand, too tight constraints do not change the feasible region much, and hence, might not yield desired change in the individual objectives.

One fundamental approach for solving the constrained optimization problem in Eq. (4) is to restate the objective function to be optimized in the so-called augmented Lagrangian function form [19]

$$L^A(\Lambda, \lambda, \rho) = f_k(\Lambda) + \sum_{p \neq k} \psi_p(\Lambda, \lambda_p, \rho_p), \quad (5)$$

where $\lambda$ and $\rho$ are the Lagrange multipliers that correspond to the upper bounds $\bar{f}_p$ and $\gamma_k$, respectively.
where $\lambda_p > 0$ and $\rho_p \geq 0$ are the Lagrange multiplier and penalty parameter, respectively, associated with the $p^{th}$ objective. The superscript $k$ in $L_1^k(.)$ emphasizes the fact that the primary objective function that is being minimized is $f_k$ with all remaining objectives being constrained. As more details can be found in [20], the quantities $\psi_p(\Lambda, \lambda, \rho)$ are given by:

$$\psi_p(\Lambda, \lambda, \rho) = \begin{cases} -\lambda_p(\nabla f_p - \nabla f_p(\Lambda)) + \frac{\rho_p}{2}(\nabla f_p - \nabla f_p(\Lambda))^2, & \text{if } \nabla f_p - \nabla f_p(\Lambda) \leq \frac{\rho_p}{2} \\ -\frac{(\lambda_p)^2}{2\rho_p}, & \text{otherwise} \end{cases}$$

(6)

It is known from the theory of constrained nonlinear optimization that the numerical minimization of $L_1^k(\Lambda, \lambda, \rho)$ at the $s^{th}$ iteration requires finding a direction of descent, $d_s$, and an appropriate step-size, $\alpha_s$, in this direction [19]–[21]. The quasi-Newton direction, given by $d_s = -\nabla \omega L_1^k(\Lambda)$, is used where $H$ is an approximation to the inverse of the Hessian matrix, $\nabla^2 \omega L_1^k$ that is computed by the BFGS method [19]. The step-size is searched by the Armijo rule, which suggests to start with a large step-size in the direction of $d_s$ and try a smaller step-size as long as the objective is not reduced [20]. It is essential in ICO to judge the acceptability of the step-size found by the Armijo rule. This is because a step-size that yields a good reduction in $f_k$ may come at the expense of an unacceptable increase in some other objectives. This issue will be examined more carefully in Section IV-B. Once the direction, $d_s$, and an appropriate step-size, $\alpha_s$, are computed, the classifier parameters are updated as $\omega_{m,t+1} = \omega_{m,t} + \alpha_s d_s$.

The augmented Lagrangian method requires that any deviation from the constraint bounds are prevented by the penalty parameter. In cases where the directional step-size yields an unacceptable deviation, the penalty parameter, $\rho$, is increased (say, by 10 times) and another step-size is searched for, as devised by Powell’s method [21] to ensure global convergence.

**A. Generating the Constraints**

Starting with an iterate $\Lambda^{(1)}$ and the corresponding objective vector $f^{(1)} = (f_1^{(1)}(\Lambda), ..., f_K^{(1)}(\Lambda))$, the goal in ICO is to move into another iterate $\Lambda^{(2)}$ yielding an objective function vector $f^{(2)}$ where at least one objective function $f_k, 1 \leq k \leq K$ attains a considerably improved value, while others are possibly slightly degraded. Consider the illustration in Fig. 1 for a two-objective optimization problem, where point $A$ is the starting point, and a better compromise solution is being searched for. Due to the conflicting nature of two objectives, it is possible to achieve a reduction in $f_1$ when $f_2$ is allowed to slightly increase. To compensate for the performance loss in $f_2$, $f_1$ is allowed to slightly increase, and the best $f_2$ for the given $f_1$ value is searched for. It is important to note that, in general, it is not necessarily correct that the resulting individual objectives are preferable to the initial ones, especially when there are many competing objectives. This is because what is gained in one iteration can quickly be lost in the subsequent iterations. However, the step-size control strategy of Section IV-B promises that the new solutions are at least as preferable as the initial compromise solutions.

![Fig. 1. A close neighborhood of the current compromise solution is searched for a better compromise solution in each iteration. Each intermediate step yields an iterate on a more favorable indifference curve.](image)

Note that the vector of objective functions in one iteration is slightly perturbed, and set as the constraint bounds in the next iteration. One technique that worked well in our experiments was as follows:

$$\overline{f} = f + \delta.$$  (7)

where $f$ is the vector of objective function values obtained on the training data set, and $\delta \in \mathbb{R}^K$ is a vector of small perturbations added to $f$. Eq. (7) formulates that the remaining objectives are allowed to slightly increase from their most recent values.

**B. Finding and Validating the Step-Size**

The step-size control strategy is motivated by the fundamental concepts known as preference and indifference of utility theory [9]. Although it is beyond the scope of this paper, the principle behind utility theory is quite simple: It is to assign to each element of a set a real number in such a way that the higher the number, the more preferred the element. In utility theory, an individual’s preferences are represented by a utility function, $u(.)$. Consider three elements $z_1, z_2$ and $z_3$ in a set. When the individual prefers element $z_2$ to element $z_1$, it is said that $z_2$ is preferred to $z_1$, which is denoted with the $\succ$ symbol as in Eq. (8). In this case, the utility of $z_2$ is greater than that of $z_1$. When the two elements $z_2$ and $z_3$ are equally preferable or an individual does not have any preference for one of them, then it is said that $z_2$ and $z_3$ are indifferent. This latter relation is denoted with a $\sim$ symbol as in Eq. (9).

In this case, the utilities of $z_2$ and $z_3$ are the same. A model of the individual’s preference logic can then be constructed using $u(.)$ in the following way:

$$z_2 \succ z_1 \text{ iff } u(z_2) > u(z_1),$$  \hspace{1cm} (8)

$$z_2 \sim z_3 \text{ iff } u(z_2) = u(z_3).$$  \hspace{1cm} (9)

When the negative of a cost function is chosen as a utility function, the point where the cost function attains a lower value is of better utility. This situation is illustrated in Fig. 2, where $z_2$ and $z_3$ are of the same utility and equally preferable, and both are of better utility than $z_1$.

When the number of objectives increases, the major difficulty is the necessity of being able to control how much the individual objectives change in each update iteration. With the increasing number of objectives, it becomes essential to
control changes from one iteration to the next so that what is gained in one iteration is not lost in the following iteration. The step-size control strategy of ICO is developed only to evaluate local changes in objective functions, and is described next.

Let the current objectives be $f_1^{(t)}$ and $f_2^{(t)}$. Without loss of generality, assume that we are solving the subproblem where $f_1$ is being minimized and $f_2$ has a constraint on it. We allow $f_2$ to slightly increase from its current value to a value $f_2^{(t+1)} = f_2^{(t)} + \delta_2, \delta_2 > 0$ with the intention that $f_1$ will reduce substantially from its current value to a value $f_1^{(t+1)} = f_1^{(t)} - \delta_1, \delta_1 > 0$. All we want is that the increase in $f_2$ is compensated by the reduction in $f_1$, i.e., $\delta_1 > \delta_2$, in fact we want to have $\delta_1 \gg \delta_2$. In either case, this means that the sum $(f_1^{(t)} + f_2^{(t)})$ is greater than the sum $(f_1^{(t+1)} + f_2^{(t+1)})$

$$(f_1^{(t)} + f_2^{(t)}) = f_1^{(t)} - \delta_1 + f_2^{(t)} + \delta_2$$

$$= [f_1^{(t)} + f_2^{(t)}] - [\delta_1 - \delta_2],$$

$$\delta_1 - \delta_2 > 0$$

Hence, the next iteration provides better objective values if the sum $(f_1 + f_2)$ reduces to a smaller value. This fact is our motivation to use average error rate as the utility function to judge the preferability of the new step found with the Armijo rule. That is, within the ICO framework, the step-size found by Armijo rule is validated and the classifier models are updated only in those cases which do not result in degradation in terms of average error.

C. Algorithm Description

In this section, we present an algorithmic implementation of the proposed ICO framework. There are two main algorithms, namely the ICO-K and MinimizeOneObjective algorithms. These two algorithms are summarized in Tables II and III.

The ICO-K algorithm minimizes $f_k$ subject to constraints on other objective functions, i.e., $f_p \leq \mathcal{T}_p, p \neq k, p = 1, ..., K$. This in turn requires updating the parameters of all $M$ classifiers. This is achieved by the MinimizeOneObjective algorithm in each iteration of which the goal is to first find the quasi-Newton direction, $d_t$, that satisfies the so-called descent condition, $[\nabla_w L_k]_w d_t < 0$ [19]. Upon finding $d_t$, an appropriate step-size $\alpha_t$ in this direction is searched for. The step-size should satisfy two conditions. First of all, any deviation from the constraints should be negligibly small. Otherwise, the penalty parameters, $\rho_p$’s, are increased and another step-size is searched for. Secondly, the step-size is validated only if the pre-chosen utility function $u$ reduces or stays the same when the step is taken. Once a step-size that satisfies both of these conditions is found, the classifier parameters are updated. This is followed by the update of the approximation of inverse Hessian, $H_t$, and the Lagrange parameters, $\lambda_p$’s [19].

D. Fair Comparison with Other Techniques

In this section, we address the issue of comparing the MOP-trained classifiers with the more traditional SOP-trained classifiers. Since all of the conflicting objectives are important to a system designer, no single overall performance measure should be taken as a basis for a realistic comparison. For many realistic classification tasks, we typically want to prevent bias towards any of the objectives and desire a symmetry across the many competing objective functions. We would hardly accept to sacrifice one objective severely in exchange of improving the other(s).

A realistic evaluation of classifiers can be achieved by comparing the range of values that the individual objective functions take on. ICO-trained classifiers offer flexibility to design a system to yield a narrower range for the individual objectives than the SOP-trained classifiers. A narrow range of objective function values mean that each of the objective function has taken reasonable consideration in the optimization process so that all objectives attain a somewhat comparable values. On the contrary, a wide range of values mean some objective functions have stolen from the share of some other objectives. As a corollary, the ICO framework is promising for producing less outliers in the objective function space compared to the SOP algorithms with an overall objective function. This is especially important when there are more competing objectives than just two. One way to quantify the degree to which a classifier results in outlier objective values is to compare the upper and lower 5% or 10% percentile averages.

V. AN APPLICATION: LID

National Institute of Standards and Technology (NIST) has sponsored several language recognition evaluations (LREs) in 1996, 2003 and 2005 [22]. The goal is to establish the baseline of current performance capability for language recognition of conversational telephone speech, and to lay the groundwork for further research efforts in the field.

A. Evaluation of LID Systems

According to the NIST evaluation specification, the system to be evaluated must determine whether or not the speech is from the target language given a test segment of speech and a target language. The speech segments contain three nominal durations of speech, namely 3 seconds, 10 seconds and 30 seconds.
The performance of an LID system is to be evaluated, first, by a detection cost function, which reduces to the average error rate, $E_{avg}$ given by Eq. (1). Further NIST evaluations involve a detailed examination of language effects in recognition performance [23]. This examination is made on the basis of the individual error rates of each language separately. The detection error tradeoff (DET) curves [24], which are functionally similar to receiver operating characteristics (ROC) curves [25], for each language are reported to serve to this purpose. Also recorded are the equal-error rate (EER), which is a continuous function of the classifier parameters, and attempts to emulate the cost of this decision, a class-specific misclassification function, $\pi_m(x, \Lambda)$, is assigned to each sample, $x \in C_m$, as

$$\pi_m(x, \Lambda) = - g(x, \omega_m) + \log \left[ \frac{1}{M-1} \sum_{i \neq m} \exp(\eta g(x, \omega_i)) \right]^{\frac{1}{\eta}},$$  \hspace{1cm} (14)$$

where $\eta$ is a positive number [27], [28], and $\Lambda$ denotes the collection of the LDF weight vectors. By varying the value of $\eta$, one can take all the competing classes into consideration when searching for the classifier models. When $\eta$ approaches $\infty$, the second term in Eq. (14) becomes $\max_{j, i \neq m} g(x, \omega_j)$.

$$\pi_m(x, \Lambda) = \text{a continuous function of the classifier parameters},$$

and attempts to emulate the decision rule. For an $m^{th}$ class sample, $x \in C_m$, $\pi_m(x, \Lambda) > 0$ implies misclassification, and $\pi_m(x, \Lambda) \leq 0$ means correct classification. In order to approximate a given objective criterion as a function of $M$ classifier models, the misclassification measure of Eq. (13) is embedded in a smoothed zero-one function such as a sigmoid function $f_k$.

$$f_k(x, \omega_k, \Lambda) = \text{a function of all of the LDF weight vectors. By varying the value of } \eta, \text{ one can take all the competing classes into consideration when searching for the classifier models. When } \eta \text{ approaches } \infty, \text{ the second term in Eq. (14) becomes } \max_{j, i \neq m} g(x, \omega_j).$$

The LDF weight vectors are solutions of the following optimization problem:

$$\min_{\omega_k} f_k(x, \omega_k, \Lambda) \text{ subject to } f_k(\omega_k) \leq \bar{T}_p, p \neq k,$$

where $\bar{T}_p$ is a function of the classifier parameters, and $f_k$ is an objective function that has the worst performance on the development data is the first to be optimized, and so on. Since each $f_k$ is a function of all of the $M$ class-discriminant functions, we need to update all $\omega_k$. Hence, for each $k = 1, ..., K$:

a. Choose $f_k$ to be minimized.

b. **Repeat for each** $p = 1, ..., K$:

   i. Call MinimizeOneObjective algorithm to update $\omega_k$.

ii. Classify the data to get $f_k$.

iii. Perturb $f_k$ to update $\tilde{f}_k$ vector.

The subscripting $(k)$ refers to Step 2, where the objective functions are ranked in decreasing order. This means that the objective function that has the worst performance on the development data is the first to be optimized, and so on. Since each $f_k$ is a function of all of the $M$ class-discriminant functions, we need to update all $\omega_k$. Hence, for each $k = 1, ..., K$:

a. Choose $f_k$ to be minimized.

b. **Repeat for each** $p = 1, ..., K$:

   i. Call MinimizeOneObjective algorithm to update $\omega_k$.

ii. Classify the data to get $f_k$.

iii. Perturb $f_k$ to update $\tilde{f}_k$ vector.

where $\omega_k^T$ represents the transpose of the weight vector of the $m^{th}$ class. An unseen sample, $\tilde{x}$, is assigned to the class $\hat{C}$ for which $g(\tilde{x}, \omega_j)$ is maximized, i.e.,

$$\hat{C} = \arg \max_j g(\tilde{x}, \omega_j).$$  \hspace{1cm} (13)$$

Let $C_m$ denote the set of training samples that are labeled as samples of the $m^{th}$ language. Within the MCE framework, in order to simulate the cost of this decision, a class-specific misclassification function, $\pi_m(x, \Lambda)$, is assigned to each sample, $x \in C_m$, as

$$\pi_m(x, \Lambda) = - g(x, \omega_m) + \log \left[ \frac{1}{M-1} \sum_{i \neq m} \exp(\eta g(x, \omega_i)) \right]^{\frac{1}{\eta}},$$

where $\eta$ is a positive number [27], [28], and $\Lambda$ denotes the collection of the LDF weight vectors. By varying the value of $\eta$, one can take all the competing classes into consideration when searching for the classifier models. When $\eta$ approaches $\infty$, the second term in Eq. (14) becomes $\max_{j, i \neq m} g(x, \omega_j)$.

$$\pi_m(x, \Lambda) = \text{a continuous function of the classifier parameters},$$

and attempts to emulate the decision rule. For an $m^{th}$ class sample, $x \in C_m$, $\pi_m(x, \Lambda) > 0$ implies misclassification, and $\pi_m(x, \Lambda) \leq 0$ means correct classification. In order to approximate a given objective criterion as a function of $M$ classifier models, the misclassification measure of Eq. (13) is embedded in a smoothed zero-one function such as a sigmoid

\begin{equation}
\tilde{f}_k(x) = \frac{1}{1 + \exp(-\beta_k (f_k(x, \omega_k, \Lambda) - \bar{T}_p))},
\end{equation}

where $\beta_k$ is a positive number, and $\bar{T}_p$ is a function of the classifier parameters, and $f_k$ is an objective function that has the worst performance on the development data is the first to be optimized, and so on. Since each $f_k$ is a function of all of the $M$ class-discriminant functions, we need to update all $\omega_k$. Hence, for each $k = 1, ..., K$:

a. Choose $f_k$ to be minimized.

b. **Repeat for each** $p = 1, ..., K$:

   i. Call MinimizeOneObjective algorithm to update $\omega_k$.

ii. Classify the data to get $f_k$.

iii. Perturb $f_k$ to update $\tilde{f}_k$ vector.
A. LID Data

which collectively classify unseen samples to the above list of primary task, the system designer needs to build 15 classifiers, dialects of Spanish, Vietnamese.

approximate the FR and FA rates for the error count. Using the loss function of Eq. (14), we can penalty which becomes essentially a classification/recognition provided two test sets, one in 1996 and one in 2003. The total size of the training set is 177,229 samples. NIST languages and dialects coming from the CallFriend corpus [29].

language. The target languages and dialects are to determine whether or not the speech is from the target LID task described in Section V. Given a speech segment

\[ \pi \text{ implies correct classification, virtually no loss is incurred.} \]

\[ \text{Clearly, when } \pi \text{ is (negative and) very small, which implies correct classification, virtually no loss is incurred.} \]

\[ \text{When } \pi \text{ is (positive and) very large, it leads to a penalty which becomes essentially a classification/recognition error count. Using the loss function of Eq. (14), we can approximate the FR and FA rates for the } m^{th} \text{ category as} \]

\[ E^{FR}_m(\Lambda) \approx \frac{1}{|C_m|} \sum_{x \in C_m} \left[ 1 - \ell_m(x, \Lambda) \right], \]

\[ E^{FA}_m(\Lambda) \approx \frac{1}{|S| - |C_m|} \sum_{x \notin C_m} \ell_m(x, \Lambda). \]

where \(|.|\) denotes the size of a set.

VI. EXPERIMENTS AND DISCUSSIONS

In this section, we report our experimental results on the LID task described in Section V. Given a speech segment of duration 30 second and a target language, the task is to determine whether or not the speech is from the target language. The target languages and dialects are Hindi, Arabic, Japanese, Farsi, Tamil, French, German, Vietnamese, Korean, two dialects of Mandarin, two dialects of Spanish and two dialects of English. For the primary task, the system designer needs to build 15 classifiers, which collectively classify unseen samples to the above list of 12 main languages, no dialect identification is required.

A. LID Data

The training data set was organized for a total of 15 languages and dialects coming from the CallFriend corpus [29]. Each language or dialect has roughly 11,000 training samples. The total size of the training set is 177,229 samples. NIST provided two test sets, one in 1996 and one in 2003. The 1996 test set has a total of 1492 samples, where each language has roughly 80 samples except for English, which has 478 samples. The 2003 test set has a total of 1280 samples: 80 samples for each language except for English and Japanese, which have 240 samples and 160 samples, respectively. We used the 2003 test data as the development set and the 1996 test data as the test set in this work.

B. Score Distribution Feature Vector

Instead of using frame-based vectors as the front end features in most conventional LID systems, we extract utterance-based score vectors generated by parallel phone recognizers followed by language models (P-PRLM) [10] and bag of sounds (BOS) [30] models. Seven phone recognizers were built: English, Korean, Mandarin, Japanese, Hindi, Spanish and German. English phonemes are trained from IIR-LID corpus 2. Korean phonemes are trained from LDC Korean corpus (LDC2003S03). Mandarin phonemes are trained from the MAT corpus [31]. Other phonemes are trained from OGI-TS corpus 3. 39-dimensional mel frequency cepstral coefficients (MFCC) features are extracted from each frame. Utterance based cepstral mean subtraction is applied to the MFCC features to remove channel distortion. Each phoneme in the languages are modeled with a hidden Markov model(HMM) of 3-state. First, the 16-language/dialect training database is tokenized into a collection of text-like phone sequences from each of the 7 tokenizers. We compute P-PRLM scores based on the resulting phone sequences. This way, we train up to 3-gram phone language model (LM) for each P-PRLM tokenizer-target language pair, resulting in 16*7=112 LMs. For each input utterance, 112 interpolated scores were derived to form a vector. In this way, all training utterances can be represented

1The NIST 2003 LRE test set also includes 80 samples for the Russian language that represents an additional challenge because no training data is provided for Russian.

2Available at http://sdp.i2r.a-star.edu.sg.

3Available at http://cslu.cse.ogi.edu/corpora/corpCurrent.html.

TABLE III

<table>
<thead>
<tr>
<th>OBJECTIVE ALGORITHM</th>
</tr>
</thead>
<tbody>
<tr>
<td>I. Set } t = 0.</td>
</tr>
<tr>
<td>II. Repeat until convergence:</td>
</tr>
<tr>
<td>1. Calculate the gradient } \nabla_\omega L^k_\omega.</td>
</tr>
<tr>
<td>2. Calculate the Newton direction, } d_\ell.</td>
</tr>
<tr>
<td>3. We want to find a step-size } \alpha_\ell \text{ in the direction of } d_\ell. If such a direction is found, it means that we can improve } L^k_\omega. However, we need to validate the found step-size:</td>
</tr>
<tr>
<td>a. Check if the deviations from the constraints (if any) are acceptable. This can be achieved by calculating the norm of the slack vector }</td>
</tr>
<tr>
<td>b. Check if the new step results in a more desirable indifference curve. This can be evaluated by comparing the local utilities } u. If } u \leq u^0, then:</td>
</tr>
<tr>
<td>i. Set } u^0 = u.</td>
</tr>
<tr>
<td>ii. Update } H_t \text{ by the BFGS update rule.</td>
</tr>
<tr>
<td>iii. Update } \lambda \text{ by Powell’s method.</td>
</tr>
<tr>
<td>iv. Save the updated } \omega(\rho).</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>MINIMIZE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set } t = 0.</td>
</tr>
<tr>
<td>Repeat until convergence:</td>
</tr>
<tr>
<td>1. Calculate the gradient } \nabla_\omega L^k_\omega.</td>
</tr>
<tr>
<td>2. Calculate the Newton direction, } d_\ell.</td>
</tr>
<tr>
<td>We want to find a step-size } \alpha_\ell \text{ in the direction of } d_\ell. If such a direction is found, it means that we can improve } L^k_\omega. However, we need to validate the found step-size:</td>
</tr>
<tr>
<td>a. Check if the deviations from the constraints (if any) are acceptable. This can be achieved by calculating the norm of the slack vector }</td>
</tr>
<tr>
<td>b. Check if the new step results in a more desirable indifference curve. This can be evaluated by comparing the local utilities } u. If } u \leq u^0, then:</td>
</tr>
<tr>
<td>i. Set } u^0 = u.</td>
</tr>
<tr>
<td>ii. Update } H_t \text{ by the BFGS update rule.</td>
</tr>
<tr>
<td>iii. Update } \lambda \text{ by Powell’s method.</td>
</tr>
<tr>
<td>iv. Save the updated } \omega(\rho).</td>
</tr>
</tbody>
</table>

function:

\[ \ell_m(x, \Lambda) = \frac{1}{1 + \exp(-\alpha \pi_m(x, \Lambda) + \beta_m)}. \]
by a collection of 112-dimension score vectors. Next, BO scores were evaluated. The BOS method uses a universal sound recognizer to tokenize an utterance into a phone sequence, which is then converted into a count vector, known as a BOS vector [30]. The universal sound inventory is a combine phoneme set from 6 languages: English, Mandarin, Japanese, Hindi, Spanish and German, a subset of the 7 languages above. There are 258 phonemes in total. For each phone sequence generated from the universal sound tokenizer, we count the occurrence of bi-phones. A phone sequence is then represented as a vector of bi-phone occurrence with 66, 564 = 258 × 25 elements. A support vector machine (SVM) is used to partition the high dimensional vector space. As SVM is a 2-way classifier, we train pair-wise SVM classifiers for the 16 target languages, resulting in 16 × 15/2 = 120 SVM classifiers. The linear kernel is adopted when using SVM-light tool. Finally, the above two sets of scores for each utterance, a vector of 112 dimensions obtained from the P-PRLM and a vector of 120 dimensions from the BOS methods, were concatenated to form the feature vector, x, of 232 dimensions for our back end system.

C. Experimental Setup

The FA and FR rates are first approximated as smooth functions of the classifier models as described in Section V-B. The approximations in Eqs. (13) and (14) require determining the right setup for the sigmoid parameters, α and β, and η. In addition, generating the constraint bounds in Eq. (7) require experimental investigation.

In our experiments, η is fixed at 7. α is fixed at 1, and β_m is heuristically adjusted for each class. The heuristics is such that the π_m(x, Λ) that are small in magnitude are summed, averaged and the resulting average is used as β_m. The idea behind the adopted heuristics is to associate more loss to those samples for which ℓ_m(d_m(x, Λ)) is close to 0.5. Such samples represent the more confusable examples to the classifier. In our experiments, for generating constraint bounds as in Eq. (7), we set δ = (10^{-3}, ..., 10^{-5}).

D. ICO with Two Objectives

One approach to solving the LID problem formulated above is to have a tradeoff between the two conflicting metrics, E^{FR}(Λ) and E^{FA}(Λ), to reach a balance for each language, i.e., to design independent binary classifiers.

In Fig. 3, the changes of E_{FR}, E_{FA} and E_{avg} for both the training and test data are illustrated separately for the language Mandarin (MA). With an inspection of the change of errors rates, we see that E_{FA} is reduced from 31.90% to 7.69% (i.e. by more than 24%) at the expense of an increase in the E_{FR} from 1.11% to 3.09% (i.e. by less than 2%) in the training data. Similarly in the test data, E_{FA} is reduced from 31.29% to 9.21% (i.e. by more than 24%) at the expense of an increase in E_{FR} from 0.00% to 1.28%. It is remarkable that the two conflicting objectives move towards a balance: To improve one objective, we allow the other objective to slightly degrade. Similar results were obtained for the other languages, and further experiments support that the improvement is obtained only with a slight degradation.

NIST have provided tools for making detection error trade-off (DET) performance plots of detection systems [24]. Using these tools, one can also compute the point on the DET curve where the average error rate is minimized. In Fig. 4, we plotted DET curves for the language Mandarin "MA" (with 156 samples out of 1492 samples) when two detection systems, one SOP-trained and one ICO-trained, were used. Furthermore, in Table IV, we report the point in the DET curve where the average of FR and FA rates was minimized. Several conclusions in favor of ICO-trained classifiers are drawn from the DET curves in Fig. 4 and the results in Table IV:

- Although the SOP classifiers were trained so that an equally-weighted sum of FR and FA rates was minimized, the individual error rates at the point where their average was minimal were highly unbalanced: The FA rate was 9.83%, while the FR rate was only 0.64%. Meantime, minimizing the average error rate over the training samples did not generalize well to the test samples.
- At the point of minimal average error rate, the ICO-trained classifiers yielded an FA rate of 5.69%, and an FR rate of 4.49%. Hence, as was our motivation in this research, ICO training was successful at achieving a good balance, or tradeoff, among the two conflicting objectives.
- As can be observed from Fig. 4, the equal error rate of the ICO-trained classifiers is also lower than that of the SOP-trained classifiers.
- From Fig. 4, we observe that when the FR rate is lower than 2%, which corresponds to classifying 3 MA samples as non-MA samples, SOP-trained classifiers performed slightly better. For all other FR rates, ICO-trained classifiers performed better. Besides, the DET curve of SOP-trained classifiers was sharper than that of ICO-trained classifiers, which suggests that when we attempt to only slightly reduce the FA rate, the FR rate degrades dramatically.
- When the system design requires a tradeoff among error types, a single performance number is inadequate to represent the capabilities of a system. This is because such a system has many operating points, and is best represented by a performance curve. Therefore, optimiza-
which are favored or sacrificed by the SOP training.

The improvements in the individual error rates when ICO, instead of SOP, was used as learning goal are noteworthy. For example, the FR rate for Arabic was reduced from 12.50% to 5.00%, and the FR rate for Hindi was reduced from 39.47% to 25.00%. The relative improvements for these two languages are 58.3% and 34.8%, respectively. On the other hand, we witness a decrease in the FR rates for Spanish and Japanese. The decreases are 37.1% and 34.4%, respectively. One important reason for this is, as already noted in Section IV-D, the outlier objective values are smoothed. This is evidenced by the upper and lower 10% percentile averages are reported in the bottom two rows of Table V. The lower 10% percentile average is lower for the SOP than for the ICO-trained classifiers, and the upper 10% percentile average is higher for SOP than for the ICO-trained classifiers. This verifies that there are some objective functions which are favored or sacrificed by the SOP training.

For clarity, we have also mapped the error rates in Table V into a portion of the real line as in Fig. 5. We have not plotted the FR and FA rates on the same graph since their scales are very different. Nevertheless, it should be kept in mind that each of the FR and FA rates compete against all others. It is depicted in Fig. 5 (a) that the major outlier FR rate was recorded for Hindi with a value of 39.47%. The ICO algorithm was able to drop it to 25% with only (relatively) small changes in the other objectives. Since any improvement in one objective comes at the expense of degradation in some other FR or FA rates, some FR and FA rates increased as expected.

As shown in Fig. 5, the FR rates for the SOP classifiers range from 2.72% to 39.47%. With ICO training, this wide range was reduced to a range from 3.75% to 25.00%. Hence, apparently, the best FR rates are increased while the worst ones are greatly reduced, i.e., outliers are smoothed. This experimental finding supports our argument that classifiers with a single training objective that is an accumulation of several conflicting objectives are not good at resulting in acceptable objective values for individual ones. In contrast, ICO targets at achieving a better balance among the competing objectives. Similar arguments are valid for FA rates as well, as shown in Fig. 5 (b).

Analyzing the Individual Error Rates

The improvements in the individual error rates when ICO, instead of SOP, was used as learning goal are noteworthy. For example, the FR rate for Arabic was reduced from 12.50% to 5.00%, and the FR rate for Hindi was reduced from 39.47% to 25.00%. The relative improvements for these two languages are 58.3% and 34.8%, respectively. On the other hand, we witness an increase in the FR rates for English, Japanese and Spanish. As already emphasized, the improvements in some objectives come at the expense of some degradation in some others. It is also noted in the last column of Table V that whenever an FR rate for a language is increased relative to the first column, its corresponding FA rate reduces and vice versa, as our analysis suggests.

### Table V

**Compromise solutions for the SOP- and ICO-trained classifiers**

<table>
<thead>
<tr>
<th>Language</th>
<th>SOP Classifiers</th>
<th>ICO Classifiers</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arabic</td>
<td>12.50 0.21</td>
<td>5.00 0.28</td>
</tr>
<tr>
<td>English</td>
<td>2.72 0.97</td>
<td>3.97 0.69</td>
</tr>
<tr>
<td>Farsi</td>
<td>10.00 0.64</td>
<td>7.50 0.85</td>
</tr>
<tr>
<td>French</td>
<td>7.50 0.50</td>
<td>3.75 0.92</td>
</tr>
<tr>
<td>German</td>
<td>6.25 0.78</td>
<td>6.25 0.85</td>
</tr>
<tr>
<td>Hindi</td>
<td>39.47 0.57</td>
<td>25.00 1.13</td>
</tr>
<tr>
<td>Japanese</td>
<td>6.33 0.71</td>
<td>10.12 0.28</td>
</tr>
<tr>
<td>Korean</td>
<td>10.26 1.84</td>
<td>10.26 0.92</td>
</tr>
<tr>
<td>Mandarin</td>
<td>4.49 0.97</td>
<td>3.85 1.20</td>
</tr>
<tr>
<td>Spanish</td>
<td>9.15 0.49</td>
<td>10.46 0.07</td>
</tr>
<tr>
<td>Tamil</td>
<td>13.70 0.49</td>
<td>12.33 0.79</td>
</tr>
<tr>
<td>Vietnamese</td>
<td>13.92 0.99</td>
<td>13.92 0.42</td>
</tr>
</tbody>
</table>

- **Average**
  - Upper 10% Percentile Avg.: 26.70% 19.46% 0.49 0.64 5.24 6.33 0.07 10.26 0
  - Lower 10% Percentile Avg.: 0.11% 0.17% 0.65 0.97 13.92 0.81 2.72 5.69 0.42 0.70 6.25 0.69 9.83 0.85 0.78 0.11% 13.70 0.17% 5.24 6.33 0.07 10.26 0

E. LID with More Than Two Objectives

Using conventional SOP techniques, $E_{avg}$ in Eq. (1) could be directly minimized. Henceforth, we refer to these classifiers as SOP-trained classifiers. In the current work, the SOP-trained classifiers are used as baseline. More details concerning the training of the SOP-trained classifiers can be found in [34].

In Table V, the individual error rates for the SOP- and ICO-trained classifiers for each language are reported. There are different ways to look at the results in Table V.

**Analyzing the Summary Statistics**

Before talking about more specifics, the summary statistics presented in the bottom few rows deserve some particular attention. Firstly, both the average FA and FR rates are significantly reduced by the ICO training with 17.6% and 16% relative improvement, respectively. One important reason for this is, as already noted in Section IV-D, the outlier objective values are smoothed. This is evidenced by the upper and lower 10% percentile averages are reported in the bottom two rows of Table V. The lower 10% percentile average is lower for the SOP than for the ICO-trained classifiers, and the upper 10% percentile average is higher for SOP than for the ICO-trained classifiers. This verifies that there are some objective functions which are favored or sacrificed by the SOP training.

For clarity, we have also mapped the error rates in Table V into a portion of the real line as in Fig. 5. We have not...
VII. CONCLUSION

It has been increasingly recognized that most realistic problems involve the simultaneous consideration of conflicting goals. The common approach is to set single overall objective function that can combine all the utilities, which in most cases overlooks the underlying tradeoffs between the conflicting objectives. Multiple objective programming offers new horizons for solving problems with conflicting objectives.

In this paper, we presented a formal formulation of an MOP approach to classifier learning, called iterative constrained optimization (ICO). The original optimization problem with conflicting objectives is formulated as an iterative process of single objective programming problems in which the remaining objectives are embedded as constraints. We refine the bounds needed to constrain the conflicting objectives in an iterative manner.

Consequently, we illustrated the use of ICO on automatic language identification where the best compromise solution among a total of 30 competing objectives was searched for. Our experimental results indicated that SOP-trained classifiers are not good at resulting acceptable objective values for individual ones. In contrast, ICO training is capable of achieving a better balance among the competing objectives. We also observed that the ICO-trained classifiers give less outlier objective values than the conventional SOP-trained classifiers.

We believe ICO is well suited for many problems in a wide range of applications. We will further this line of research in several theoretically rich directions. One of the first ones is about the automatic means to infer the constraint bounds. We have observed in our experiments that different settings for the constraint bounds directly translate into different end results, as our intuition also suggests. Based on our experience, we foresee that the constraint bounds can be set by analyzing the sensitivity of the problem on the changes of the individual objectives.

VIII. ACKNOWLEDGEMENTS

The authors thank Haizhou Li, Bin Ma, Rong Tong and Donglai Zhu for sharing their LID data for use in the experiments.

REFERENCES


Chin-Hui Lee is a professor at School of Electrical and Computer Engineering, Georgia Institute of Technology in Atlanta, Georgia. Dr. Lee received the B.S. degree in Electrical Engineering from National Taiwan University, Taipei, Taiwan in 1973, the M.S. degree in Engineering and Applied Science from Yale University, New Haven, CT., in 1977, and the Ph.D. degree in Electrical Engineering with a minor in Statistics from University of Washington, Seattle, WA., in 1981.

After graduation in 1981, Dr. Lee joined Verbex Corporation, Bedford, MA., and was involved in research on connected word recognition. In 1984, he became affiliated with Digital Sound Corporation, Santa Barbara, where he engaged in research and product development in speech coding, speech synthesis, speech recognition and signal processing for the development of the DSC-2000 Voice Server. Between 1986 and 2001, he was with Bell Laboratories, Murray Hill, New Jersey, where he became a Distinguished Member of Technical Staff and Director of the Dialogue Systems Research Department. His research interests include multimedia communication, multimedia signal and information processing, speech and speaker recognition, speech and language modeling, spoken dialogue processing, adaptive and discriminative learning, biometric authentication, information retrieval, and bioinformatics. His research scope is reflected in “Automatic Speech and Speaker Recognition: Advanced Topics”, published by the Kluwer Academic Publishers in 1996. From August 2001 to August 2002 he was a visiting professor at School of Computing, The National University of Singapore. In Sepetember 2002, he joined the Faculty of School of Electrical and Computer Engineering at Georgia Institute of Technology.

Prof. Lee has participated actively in professional societies. He is a member of the IEEE Signal Processing Society, Communication Society, Computer Society, and the International Speech Communication Association. In 1991-1995, he was an associate editor for the IEEE Transactions on Signal Processing and Transactions on Speech and Audio Processing. During the same period, he served as a member of the ARPA Spoken Language Coordination Committee. In 1995-1998 he was a member of the Speech Processing Technical Committee of the IEEE Signal Processing Society (SPS), and later became the chairman of the Speech TC from 1997 to 1998. In 1996, he helped promote the SPS Multimedia Signal Processing (MMSP) Technical Committee in which he is a founding member.

Dr. Lee is a Fellow of the IEEE, and has published more than 300 papers and 25 patents on the subject of automatic speech and speaker recognition and multimedia information Processing. He received the SPS Senior Award in 1994 and the SPS Best Paper Award in 1997 and 1999, respectively. In 1997, he was awarded the prestigious Bell Labs President’s Gold Award for his contributions to the Lucent Speech Processing Solutions product. In 2006, he received the IEEE signal Processing Society’s Technical Achievement Award. Dr. Lee is also a frequent invited speaker in international communities. In 2000, he was named one of the six Distinguished Lecturers by the IEEE Signal Processing Society. In 2007, he was named one of the two inaugural Distinguished Lecturers for the International Speech Communication Association.

Sibel Yaman is a Ph.D. student in School of Electrical and Computer Engineering at Georgia Institute of Technology, Atlanta, GA. She received her B.S. degree in electrical and electronic engineering from Bilkent University, Ankara, Turkey in 2002, and her M.S. degree from Georgia Institute of Technology, Atlanta, GA in 2004. She is a recipient of the Microsoft Research Graduate Fellowship for 2006-2007. She has been selected as a Best Student Paper Award finalist in ICASSP 2006. Her research interests include text categorization, automatic language identification, multi-objective optimization techniques for pattern classification, and language modeling.