5

Similarity Discriminant Analysis

Luca Cazzanti
Applied Physics Lab
Box 355640
University of Washington, Seattle, WA 98105,
USA

1. Introduction

This chapter details *similarity discriminant analysis* (SDA), a new framework for similarity-based classification. The two defining characteristics of the SDA classifica- tion framework are *similarity-based* and *generative*. The classifiers in this framework are similarity-based, because they classify based on the pairwise similarities of data samples, and they are generative, because they build class-dependent probability models of the similarities between samples. Similarity-based classifiers already exist; classifiers based on generative models already exist. SDA is a *new* framework for classification comprising classifiers that are *both* similarity-based and generative.

Within the general SDA framework, this chapter describes several families of classifiers: the *SDA classifier*, the *local SDA classifier*, and the *mixture SDA classifier*. The SDA classifier is at the foundation of SDA. It classifies based on the class-conditional generative models of the similarity of the samples to representative class prototypes, or *centroids*. The SDA framework is introduced, developed, and discussed with the aid of this centroid-based SDA classifier. Then, the centroid-based SDA classifier is generalized beyond class centroids to arbitrary class-descriptive statistics. Other possible statistics are described, illustrating the power and generality of the SDA framework.

The local SDA classifier is a local version of the SDA classifier. It builds similarity-based class-conditional generative models within a neighborhood of a test sample to be classified. The local class models are endowed with low bias and retain the powerful quality of interpretability associated with generative probability models. Local SDA is a consistent classifier, in the sense that its error rate converges to the Bayes error rate, which is the best possible error rate attainable by a classifier.

The mixture SDA classifier draws from the well-established metric learning mixture model research. It generalizes the single-centroid SDA classifier to a mixture of single-centroid SDA components. The mixture SDA classifier can be trained with an expectation-maximization (EM) algorithm which parallels the standard EM approach for the well-known Gaussian mixture models.

The problem of classifying samples based only on their pairwise similarities may be divided into two sub-problems: measuring the similarity between samples and classifying the samples based on their pairwise similarities. It is beyond the scope of this chapter to discuss exhaustively and in detail various ways to measure similarity and various similarity-based

Source: Machine Learning, Book edited by: Abdelhamid Mellouk and Abdennacer Chebira,
ISBN 978-3-902613-56-1, pp. 450, February 2009, I-Tech, Vienna, Austria

classifiers. The reader is referred to the references for more details; here, only a brief summary of relevant techniques is provided

1.1 Measuring similarity

Judging similarity between samples characterized by many disparate data types poses challenges of data representation and quantitative comparison. For example, modern databases store information from disparate data sources in different formats: multimedia databases store audio, video and text data; proteomics databases store information on proteins, genetic sequences, and related annotations; internet traffic databases store mouse click histories, user profiles, and marketing rules; homeland security databases may store data on individuals and organizations, annotations from intelligence reports, and maritime shipping records. These database objects, or samples, are described by both numerical and non-numerical data. For example, a security database might store cell phone records in textual form and voice parameters for speaker recognition in numerical form. Representing all these different data types with continuous-valued numbers in a geometric feature space is not appropriate. Thus, current metric space classifiers which rely on metric similarity functions may not be applicable.

Furthermore, in some applications, only the pairwise similarities may be observed, and the underlying features may be inaccessible. For example, one of the datasets discussed in this chapter consists of human-judged similarities between pairs of sonar echoes. For this dataset, the putative perceptual features from which the human similarity ratings are generated are unknown - indeed eliciting the features remains an ongoing research problem (Philips et al., 2006) - but the similarity ratings are nonetheless successfully used for classification. In many applications, the similarity relationship between samples may lack the metric properties usually associated with distance (minimality, symmetry, triangle inequality); thus, using a metric function to express the pairwise similarities is suboptimal. Similarities are more general than distances and require more general functions than metrics (Tversky, 1977). Several researchers have addressed the problem of measuring similarity by rpoposing several simialrity measures. Psychologists, leacd by Tversky, have proposed models of similarity that take into account context and the non-metric way in which humans judge the similarity between complex objects (Tversky, 1977; Tversky & Gati, 1978; Gati & Tversky, 1984; Sattath & Tversky, 1987). The value difference metric (VDM) was originally designed with the goal of improving nearest-neighbor classification (Stanfill & Waltz, 1986) of text documents, and subsequent improvements extended it to classification of objects characterized by both textual and numerical features (Wilson & Martinez, 1997; Cost & Salzberg, 1993). Lin proposed an information-theoretic similarity (Lin, 1998) for document retrieval; (Cazzanti & Gupta, 2006) proposed the *residual entropy similarity* measure by extending Tversky's psychological similarity models with information-theoretic notions, and showed that it strongly takes into account the context in which the similarity is being evaluated. More comprehensive reviews of similarity measures appear in (Santini & Jain, 1999) and (Everitt & Rabe-Hesketh, 1997).

1.2 Similarity-based classifiers

Similarity-based classifiers are defined as those classifiers that require only a pairwise similarity - a description of the samples themselves is not needed. Similarity-based classifiers classify test samples given a labeled set of training samples, the pairwise

similarity values of the training samples, and the similarity of the test sample to the training samples. If the description of the samples in terms of feature vectors is available, an existing or ad hoc similarity function that maps any two samples to a similarity value may be used (Bicego et al., 2006; Pekalska et al., 2001; Jacobs et al., 2000; Hochreiter & Obermayer, 2006).
Among the existing similarity-based classifiers, the simplest method is the nearest neighbor classifier, which determines the most similar training sample z to the test sample x, and classifies x as z's class:

$$\hat{y} = \arg \max_{h=1,\ldots,G} \left(\max_{z \in \mathcal{X}_h} s(x, z) \right), \tag{1}$$

where $\mathcal{X}_h$ is the set of training samples from class h. More generally, the k-nearest neighbor classifier (k-NN) determines a neighborhood of k most similar training samples to the test sample x, and classifies x as the most-frequently occurring class label among the neighbors. Experiments have shown that nearest neighbors can perform well on practical similarity-based classification tasks (Cost & Salzberg, 1993; Pekalska et al., 2001; Simard et al., 1993; Belongie et al., 2002). For example, nearest neighbor classifiers using a tangent distortion metric and a shape similarity metric have both been shown to achieve very low error on the MNIST character recognition task.
Condensed near-neighbor strategies replace the set of training samples for each class with a set of prototypes for that class. Usually the prototype set is an edited set of the original training samples (also called edited nearest neighbors), but the prototypes do not need to be from the original training set. Let c_h be the number of the prototypes $\{\mu_{hl}\}$ for class h; then, the condensed nearest neighbor rule is to classify a test sample x as the class of the prototype to which it is most similar,

$$\hat{y} = \arg \max_{h=1,\ldots,G} \left(\max_{l=1,\ldots,c_h} s(x, \mu_{hl}) \right) \tag{2}$$

Many authors have considered strategies for condensing near-neighbors for similarity-based classification to increase classification speed, decrease the required memory, remove outliers, and possibly attain better performance (Weinshall et al., 1999; Jacobs et al., 2000; Lam et al., 2002; Pekalska et al., 2006; Lozano et al., 2006). A well-known strategy for condensing nearest neighbors in non-metric spaces is the *k-medoids* algorithm (Hastie et al., 2001). Given a set of c_h candidate prototypes selected from $\mathcal{X}_h$, the remaining training samples $z \in \mathcal{X}_h$ are assigned to their nearest (most similar) prototype, so that the set $\mathcal{X}_h$ of all training samples from class h is partitioned in c_h mutually-exclusive subsets $\{\mathcal{X}_{hl}\}$, and each $\mathcal{X}_{hl}$ is uniquely associated with candidate prototype μ_{hl}. Then, the lth prototype for the hth class is updated according to the standard maximum similarity update rule, which selects the new μ_{hl} as the training sample in $\mathcal{X}_{hl}$ which is most similar to all other samples in $\mathcal{X}_{hl}$,

$$\mu_{hl}^{*} = \arg \max_{\mu_{hl} \in \mathcal{X}_{hl}} \sum_{z \in \mathcal{X}_{hl}} s(z, \mu_{hl}). \tag{3}$$

The training samples are then reassigned to the updated prototypes, and the update rule (3) is repeated. The reassignment and update steps are repeated until a predetermined

maximum number of iterations is reached or until the updated prototypes $\mu^*_{hl} = \mu_{hl}$ for all h and l. The number of prototypes in each class c_h is determined by cross-validation; the initial prototypes $\{\mu_{hl}\}$ are selected randomly from the training set.

An extreme form of condensed near-neighbors is to replace each class's training samples by one prototypical sample, often called a *centroid*. The resulting nearest centroid classifier can be considered a simple parametric model (Weinshall et al., 1999), though it lacks a probabilistic structure. Let $s(x, z)$ be the similarity between a sample x and a sample z, and let there be a finite set of classes 1, 2, ... ,G. The nearest centroid approach classifies x as the class

$$\hat{y} = \arg \max_{h=1,\ldots,G} s(x, \mu_h), \tag{4}$$

where μ_h is the representative centroid for the class h. A standard definition for the centroid of a set of training samples is the training sample that has the maximum total similarity to all the training samples of the same class (Weinshall et al., 1999; Jacobs et al., 2000):

$$\mu_h = \arg \max_{\mu \in \mathcal{X}_h} \sum_{z \in \mathcal{X}_h} s(z, \mu). \tag{5}$$

A variation of the nearest centroid classifier is the local nearest centroid classifier, which is an analog to the local nearest means classifier proposed by Mitani and Hamamoto (Mitani & Hamamoto, 2006, 2000). In this variant, the class centroids (5) are computed from a local neighborhood of each test point x; they are not computed from the entire training set. The neighborhood may be defined in many ways. The most common definition is the k-nearest neighbors. In this case, local nearest centroid is like the k-NN classifier, except that it classifies x as the class of its nearest centroid where the centroids are computed from the k-nearest neighbors of x.

The nearest centroid classifier is analogous to the nearest-mean classifier in Euclidean space, which is the optimal Euclidean-based classifier if one assumes that the class-conditional distributions are Gaussian, the class priors are equal, and that each class covariance is the identity matrix (Duda et al., 2001; Hastie et al., 2001).

2. Similarity discriminant analysis

In standard metric learning, quadratic discriminant analysis (QDA) is a generative classifier that generalizes the nearest-mean classifier by modeling each class-conditional distribution as a Gaussian (Duda et al., 2001). Analogously, SDA is a generative similarity-based classifier that generalizes the nearest-centroid classifier (Weinshall et al., 1999) by modeling each class-conditional distribution with a parametric probability model (Cazzanti et al.; Gupta et al., 2007). The SDA class-conditional probability models have exponential form, because they are derived as the maximum entropy distributions subject to constraints on the mean similarities of the data to the class centroids. As with other parametric approaches to classification, the resulting log-linear SDA classifier is powerful when it effectively models the true generating distribution. This section introduces SDA and shows how it classifies; then, it extends SDA from using class centroids to using arbitrary descriptive statistics to discriminate between the classes, including continuous-valued statistics.

2.1 A generative centroid-based classifier

Assume a class centroid μ_h has been determined for the hth class, where $h = 1, ..., G$. A problem with the nearest centroid classifier given in (4) is that it does not take into account the variability of the similarities to the centroid within a class. To take into account this variability, first consider a simple generalization of nearest centroid, here called the *adjusted nearest centroid classifier* : classify a test sample x as class $\hat{y}$ where

$$\hat{y} = \arg \max_{h=1,\ldots,G} \frac{s(x, \mu_h)}{\bar{s}_{hh}}, \tag{6}$$

and where $\bar{s}_{hh}$ is the average similarity of class h samples to the class h centroid,

$$\bar{s}_{hh} = \frac{1}{n_h} \sum_{z \in \mathcal{X}_h} s(z, \mu_h),$$

where $n_h = |\mathcal{X}_h|$. The adjusted nearest centroid classifier is analogous to the one-dimensional Gaussian rule of classifying based on the the variance-weighted distances to the class means, $\|x - \tilde{\mu}_h\| / \tilde{\sigma}_h$, where $x, \tilde{\mu}_h, \tilde{\sigma}_h \in \mathbb{R}$. The adjusted nearest centroid classifier is more flexible than the nearest centroid classifier, but lacks a probabilistic structure, and takes into account only the similarity of a sample to one class centroid.

Thus, a generative centroid-based classifier that models the probability distribution of the test sample similarity statistics $s(x, \mu_h)$ for each h is proposed. Begin with the Bayes classifier (Hastie et al., 2001), which assigns a test sample x the class $\hat{y}$ that minimizes the expected misclassification cost,

$$\hat{y} = \arg \min_{f=1,\ldots,G} \sum_{g=1}^{G} C(f, g) P(Y = g|x), \tag{7}$$

where $C(f, g)$ is the cost of classifying the test sample x as class f if the true class is g and $P(g \mid x)$ is the probability that sample x belongs in class g. In practice the distribution $P(g \mid x)$ is generally unknown, and thus the Bayes classifier of (7) is an unattainable ideal.

Assume that all test and training samples come from some abstract space of samples $\mathcal{B}$, which might be an ill-defined space, such as $\mathcal{B}$ is the set of all amino acids, or $\mathcal{B}$ is the set of all terrorist events, or $\mathcal{B}$ is the set of all women who gave birth to twins. Let $x, \mu_h, z \in \mathcal{B}$, and let the similarity function be some function $s : \mathcal{B} \times \mathcal{B} \rightarrow \Omega$, where $\Omega \subset \mathbb{R}$. If the set of possible samples $\mathcal{B}$ is finite, then the space of the pairwise similarities Ω will also be finite, and hence discrete. For simplicity, in this section assume that Ω is a finite discrete space. Continuous and possibly infinite spaces B, Ω are briefly discussed in Section 2.2.3.

Consider a random test sample X with random class label Y, where x will denote a realization of X. Assume that the relevant information about X's class label is captured by the set $\mathcal{T}(X)$ of G descriptive statistics

$$\mathcal{T}(X) = \{s(X, \mu_1), s(X, \mu_2), \ldots, s(X, \mu_G)\}.$$

That is, the relevant information about x is captured by its similarity to each class centroid. Under this assumption, given a particular test sample x, the classification rule (7) becomes: classify x as class $\hat{y}$ that solves

$$\arg \min_{f=1,\ldots,G} \sum_{g=1}^{G} C(f,g) P(Y = g | \mathcal{T}(x)).$$

Using Bayes rule, this is equivalent to the problem

$$\arg \min_{f=1,\ldots,G} \sum_{g=1}^{G} C(f,g) P(\mathcal{T}(x) | Y = g) P(Y = g). \quad (8)$$

Note that $P(\mathcal{T}(x) \mid Y = g)$ is the probability of seeing a particular set of similarities between the test sample x and the G class centroids $\{\mu_1, \mu_2, \ldots, \mu_G\}$ given that x is a class g sample.
Next, assume that each unknown class-conditional distribution $P(\mathcal{T}(x) \mid Y = g)$ has the same average value as the training sample data from class g. That is, given a random test sample X there will be a random similarity $s(X, \mu_h)$; constrain the class-conditional distribution $P(\mathcal{T}(x) \mid Y = g)$ such that

$$E_{P(\mathcal{T}(x)|Y=g)}[s(X,\mu_h)] = \frac{1}{n_g} \sum_{z \in \mathcal{X}_g} s(z,\mu_h), \quad (9)$$

holds for each g and h where n_g is the number of training samples of class g. Each constraint requires that the class-conditional expectation of one of the elements of $\mathcal{T}(X)$ is equal to the maximum likelihood estimate of that element given the training data. This makes for G constraints for each class-conditional distribution, for a total of $G \times G$ constraints because there are G class-conditional distributions. Given these constraints, there is some compact and convex feasible set of class-conditional distributions. A feasible solution will always exist because the constraints are based on the data.
As prescribed by Jaynes' principle of maximum entropy (Jaynes, 1982), a unique class-conditional joint distribution is selected by choosing the maximum entropy solution that satisfies (9). Maximum entropy distributions have the maximum possible uncertainty, such that they are as uniform as possible while still satisfying given constraints. Given a set of moment constraints, the maximum entropy solution is known to have exponential form (Cover & Thomas, 1991). For example, in standard metric learning, the Gaussian class-conditional distribution model used in LDA and QDA is the maximum entropy distribution given a specific mean vector and covariance matrix (Cover & Thomas, 1991).
The maximum entropy distribution that satisfies the moment constraints specified in (9) is

$$\hat{P}(\mathcal{T}(x)|Y = g) = \gamma_g e^{\left(\sum_{h=1}^{G} \lambda_{gh} s(x,\mu_h)\right)}, \quad (10)$$

where $\{\gamma_g, \lambda_{g1}, \lambda_{g2}, \ldots, \lambda_{gG}\}$ are a unique set that ensures that the constraints (9) are satisfied and that $\hat{P}(\mathcal{T}(x) \mid Y = g)$ is non-negative and normalized. Rewrite equation (10) as

$$\hat{P}(\mathcal{T}(x)|Y = g) = \prod_{h=1}^{G} \gamma_{gh} e^{\lambda_{gh} s(x,\mu_h)} \quad (11)$$

where $\prod_h \gamma_{gh} = \gamma_g$. Let

$$\hat{P}(s(x,\mu_h)|Y=g) = \gamma_{gh} e^{\lambda_{gh} s(x,\mu_h)};$$

then (11) can be written

$$\hat{P}(\mathcal{T}(x)|Y=g) = \prod_{h=1}^{G} \hat{P}(s(x,\mu_h)|Y=g).$$

That is, under the maximum entropy assumption, the joint distribution on $\mathcal{T}(x)$ is the product of the marginal distributions on each similarity statistic comprising the set $\mathcal{T}(X)$. Thus, the similarity statistics are conditionally independent given the class label under this model. Although one does not expect this conditional independence to be strictly valid, the hypothesis is that it will be an effective model, just as the naive Bayes' model that features are independent is optimistic but useful.

Substituting the maximum entropy solution (10) into (8) yields the classification rule: classify x as the class $\hat{y}$ which solves

$$\arg \min_{f=1,\ldots,G} \sum_{g=1}^{G} C(f,g) \left(\prod_{h=1}^{G} \gamma_{gh} e^{\lambda_{gh} s(x,\mu_h)} \right) P(Y=g). \tag{12}$$

To solve for the parameters $\{\lambda_{gh}, \gamma_{gh}\}$, one solves the G constraints individually for λ_{gh}. Then given $\{\lambda_{gh}\}$, the $\{\gamma_{gh}\}$ are trivially found using the normalization constraint. Solving for λ_{gh} is straightforward; for example, one uses the Nelder-Mead optimizer built into Matlab (version 15) in the `fminsearch()` function (Mat). This is the method used throughout this work. As an alternative, one may find the probability mass function with maximum entropy, subject to the constraints, without a priori knowledge that the solution is exponential.

The classifier given in (12) is termed the *similarity discriminant analysis* (SDA).

2.2 General generative models for similarity-based classification

The previous section introduced SDA for the case when the descriptive statistics are the similarities of the samples to the class centroids. This section generalizes SDA to arbitrary descriptive statistics $\mathcal{T}(x)$ which can be used to discriminate different classes and describes the resulting general generative model for classifying with arbitrary statistics.

2.2.1 Descriptive statistics

Several possibilities for the descriptive statistics $\mathcal{T}(x)$ are described below.

- Centroid Definitions - A standard centroid definition was given in (5). Another choice is to allow a class prototype that is not constrained to be a training sample,

$$\mu_h^* = \arg \max_{\mu \in \mathcal{B}} \sum_{z \in \mathcal{X}_h} s(z,\mu). \tag{13}$$

  In this case the solution μ_h^* requires a description of the entire space of possible samples $\mathcal{B}$. In practice, one may not know the entire sample space $\mathcal{B}$, only the training samples $\mathcal{X}$, so it may not be possible to calculate μ_h^*.

A third definition of a class prototype is based on Tversky's analysis of similarity-based near-neighbor relationships (Tversky & Hutchinson, 1986; Schwartz & Tversky, 1980), and takes into account the similarity-based ranks of a training sample's near-neighbors. Define the neighborhood $\mathcal{N}(z) \subseteq \mathcal{X}$ of a sample z as the set of training samples whose nearest neighbor in similarity space is z. The popularity of z is the size of its neighborhood $|\mathcal{N}(z)|$. The class centroid is the sample with the highest popularity, that is,

$$\mu_h = \arg\max_{z \in \mathcal{X}_h} |\mathcal{N}(z)|. \tag{14}$$

This centroid is the training sample that is most often the closest neighbor of the training samples in the class. Ties in popularity are broken by selecting the sample with the highest total similarity to its neighbors.

- Higher Order and Non-Centroidal Descriptive Statistics - Given a set of class centroids $\{\mu_h\}$, higher-order statistics could be used as, or added to, the set of descriptive statistics $\mathcal{T}(X)$, such as $(s(X, \mu_h) - E[s(X, \mu_h)])^2$, or cross-class statistics, such as $(s(X, \mu_h) - E[s(X, \mu_g)])^2$. Or, instead of the centroid-based statistics fs(X, μ_h)g, it might be more appropriate to use the nonparametric statistics formed by the total pairwise similarity for each class h, such that the hth descriptive statistic in test set $\mathcal{T}(X)$ is $\sum_{z \in \mathcal{X}_h} s(X, z)$.
- Nearest Neighbor Similarity - A descriptive statistic that is not centroid-based is the *nearest neighbor similarity*: a test sample's similarity to its most similar training sample. Given a sample x and the training samples $z \in \mathcal{X}$, the nearest neighbor similarity is defined

$$s_{nn}(x) = \max_{z \in \mathcal{X}} s(x, z). \tag{15}$$

The SDA classifier based on nearest neighbor similarity, denoted by *nnSDA*, may be viewed as a generalization of the similarity-based nearest neighbor classifier (1-NN) defined in 1. That classifier labels x with the same class label as its nearest neighbor without making use of any information about its similarity to such nearest neighbor. The nnSDA classifier, on the other hand, classifies x as the class of its nearest neighbor based on a probabilistic model of $s_{nn}(x)$. The probability model is computed with the mean-constrained maximum entropy approach of Section 2.1, which results in exponential solutions. In this case, the constraint is that the mean of the distribution must be the same as the empirical average of the observed nearest neighbor similarities. Denote by $s_{nn,h}(X)$ the random similarity of a random test sample X to its nearest neighbor in class h. For nnSDA, the constraint is written as

$$E_{P(\mathcal{T}(x)|Y=g)}[s_{nn,h}(X)] = \frac{1}{n_g} \sum_{z \in \mathcal{X}_g} s_{nn,h}(z), \tag{16}$$

and the classification rule becomes to classify as the class $\hat{y}$ that solves

$$\arg\min_{f=1,\ldots,G} \sum_{g=1}^{G} C(f,g) \left(\prod_{h=1}^{G} \gamma_{gh} e^{\lambda_{gh} s_{nn,h}(x)} \right) P(Y = g), \tag{17}$$

where the parameters λ_{gh} and γ_{gh} are computed with the same numerical optimization method used for SDA.

As further discussed in the next section, the SDA framework accommodates any desired set of descriptive statistics $\mathcal{T}(x)$: different similarity functions could be mixed, dissimilarities and similarities can be mixed, and so on.

2.2.2 Generative classifier from arbitrary descriptive statistics

Given an arbitrary set of M descriptive statistics $\mathcal{T}(x)$, the same reasoning of Section 2.1 produces a generative similarity-based classifier. First, the assumption is that $\mathcal{T}(x)$ is sufficient information to classify x leads to the classification rule given in (8). Second, for the mth descriptive statistic $T_m(x) \in \mathcal{T}(x)$, $m = 1, ..., M$, one assumes that its mean with respect to the class conditional distribution of $\mathcal{T}(x)$ is equal to the training sample mean:

$$E_{P(\mathcal{T}(x)|g)}[T_m(X)] = \frac{1}{n_g} \sum_{z \in \mathcal{X}_g} T_m(z). \tag{18}$$

Third, given the $M \times G$ constraints specified by (18), one estimates the class-conditional distribution to be the maximum entropy distribution,

$$\begin{aligned} \hat{P}(\mathcal{T}(x)|g) &= \prod_{m=1}^{M} \gamma_{gm} e^{\lambda_{gm} T_m(x)} \\ &= \prod_{m=1}^{M} \hat{P}(T_m(x)|g). \end{aligned} \tag{19}$$

Substituting the maximum entropy solution (19) into (8) yields the SDA classification rule: classify x as the class $\hat{y}$ which solves

$$\arg \min_{f=1,...,G} \sum_{g=1}^{G} C(f,g) P(g) \prod_{m=1}^{M} \gamma_{gm} e^{\lambda_{gm} T_m(x)} . \tag{20}$$

The parameters $\{\lambda_{gm}, \gamma_{gm}\}$ are calculated as in the centroid-based SDA case described in Section 2.1.

2.2.3 Continuous-valued statistics

The generative classification models presented in this chapter can be extended to the case in which the statistics $\mathcal{T}(x)$ are from a continuous set Ω. This will be the case, for example, when using an overlap similarity (e.g. $\max\{x[i], z[i]\}$) with real-valued features, or when the similarity between X and z is the Euclidean distance. Then, the expectation in (18) is a normalized integral over the continuous set of possible similarity values. Let a and b denote the minimum and maximum possible similarity values (and hence the lower and upper bound on the expectation's integral). Then simplifying (18) yields the relationship

$$\frac{e^{\lambda_{gm} b}(\lambda_{gm} b - 1) - e^{\lambda_{gm} a}(\lambda_{gm} a - 1)}{\lambda_{gm}(e^{\lambda_{gm} b} - e^{\lambda_{gm} a})} = \bar{t}_{gm}, \tag{21}$$

where $\bar{t}_{gm} = \frac{1}{n_g} \sum_{z \in \mathcal{X}_g} T_m(z)$. The solution to (21) can be computed numerically. For the special case $a = 0$ and $b = \infty$, the solution is $\lambda_{gm} = -1/\bar{t}_{gm}$.

3. Local SDA

This chapter introduces *local SDA* (Cazzanti & Gupta, 2007), a similarity-based classifier that is both generative and local. An advantage of generative classifiers is their interpretability: classes are modeled by conditional probability distributions which are assumed to have generated the observed data. An advantage of local classifiers it that they reduce the estimation bias problem which affects generative classifiers. Local SDA combines the qualities of both generative and local classifiers.

For the SDA classifier, the class-conditional generative distributions are exponentials that model the similarities between samples - or more generally the descriptive statistics of the sample. The exponentials are the maximum entropy distributions subject to constraints on the mean values of the similarities. However, when the underlying distributions are complex, a particular set of empirical statistics may fail to capture the necessary information about a sample's class membership. In fact, in SDA, constraining the means of the class-conditional distributions may result in too much model bias, just as the QDA model of one Gaussian per class causes model bias (Hastie et al., 2001). In standard metric learning, one way to address the bias problem while retaining the advantages of a generative approach is to form more flexible Gaussian mixture models. In similarity-based learning, mixture models may also be formed; this approach is discussed in Section 4.

Here, the bias in SDA is addressed by using local classifiers in similarity space. In metric learning, one way to avoid the bias problem is to use local classifiers, e.g. *k*-NN, which classify test samples based on the class labels of their nearest neighbors. Local classifiers do not estimate probabilistic models for the sample classes and consequently lack the interpretability of generative models. Even so, they provide an intuitive framework for classification through the concepts of nearest-neighbor and neighborhood. In this chapter, SDA is applied to a local neighborhood about the test sample. The resulting *local SDA* classifier trades-off model bias and estimation variance depending on the neighborhood size, while retaining the power of a generative classifier. To the author's knowledge, local SDA is the first example of a classifier that is both generative and local. The only arguable contender is the local nearest- mean classifier (Mitani & Hamamoto, 2000, 2006) for metric learning; however that classifier was not proposed as a generative model.

Local SDA is a straightforward variation of SDA. The local SDA classifier model is that all of the relevant information about classifying a test sample x depends only on the k nearest (most similar) training samples to x. Thus, the local SDA classifier computes the descriptive statistics from a neighborhood of a test sample. More specifically, local SDA is a log-linear generative classifier that models the probability distribution of the similarity $s(x, \mu_h)$ between the test sample x and the class centroids $\{\mu_h\}$, just like SDA. Unlike SDA, the class centroids, the class-conditional similarity probability models, and the estimates of the class priors are computed from a neighborhood of the test sample rather than from the entire training set. Thus, the class centroid definition (5) used for SDA still holds for local SDA; one simply redefines $\mathcal{X}_h$ as the subset of the k nearest neighbors from class h. The class priors are estimated using normalized class membership counts of the neighbors of x, that is $\hat{P}\ (Y = h) = |\mathcal{X}_h|/k$. The mean similarity constraints (9) for the SDA maximum entropy optimization

are formally the same for local SDA, except that the mean is computed from the neighbors of test sample x rather than the whole training set. Thus, the optimized parameters λ_{gh} and γ_{gh} are local. Given the set of local class centroids $\{\mu_h\}$, the local class priors $\hat{P}(Y = g)$, and the local class-conditional model parameters γ_{gh} the local SDA classification rule is identical to the SDA rule (12):

$$\arg \max_{f=1,\ldots,G} \sum_{g=1}^{G} C(f,g) \left(\prod_{h=1}^{G} \gamma_{gh} e^{\lambda_{gh} s(x,\mu_h)} \right) \hat{P}(Y = g).$$

A problem can occur if the hth class has few training samples in the neighborhood of test sample x. In this case, the local SDA model for class h is difficult to estimate. To avoid this problem, if the number of local training samples in any of the classes is very small, for example $n_h < 3$, the local SDA classifier reverts to the local nearest centroid classifier. If $n_h = 0$ so that $\mathcal{X}_h$ is the empty set, then the probability of class h is locally zero, and that class is not considered in the classification rule (12). This strategy enables local SDA to gracefully handle small k and very small class priors.

Local classification algorithms have traditionally been weighted voting methods, including classifying with local linear regression, which can be formulated as a weighted voting method (Hastie et al., 2001). These methods are by their nature non-parametric and their use arises in situations when the available training samples are too few to accurately build class models. On the other hand, it is known that the number of training samples required by nonparametric classifiers to achieve low error rates grows exponentially with the number of features (Mitani & Hamamoto, 2006). Thus, when only small training sets are available, nonparametric classifiers are negatively impacted by outliers. In 2000, Mitani and Hamamoto (Mitani & Hamamoto, 2000, 2006) were the first ones to propose a classifier that is both model-based and local. However, they did not develop it as a local generative method; instead, they proposed the classifier as a local weighted-distance method. Their nearest-means classifier can be interpreted as a local QDA classifier with identity covariances. In experiments with simulated and real data sets, the local nearest-means classifier was competitive with, and often better than, nearest neighbor, the Parzen classifier, and an artificial neural network, especially for small training sets and for high dimensional problems.

Local nearest-means differs from local SDA in several aspects. First, the classifier by Mitani and Hamamoto in (Mitani & Hamamoto, 2006) learns a metric problem, not a similarity problem: the class prototypes are the local class-conditional means of the features and a weighted Euclidean distance is used to classify a test sample as the class of its nearest class mean. Second, the neighborhood definition is different than the usual k nearest neighbors: they select k nearest neighbors from each class, so that the total neighborhood size is $k \times G$.

More recently, it was proposed to apply a support vector machine to the k nearest neighbors of the test sample (Zhang et al., 2006). The SVM-KNN method was developed to address the robustness and dimensionality concerns that a²ict nearest neighbors and SVMs. Similarly to the nearest-means classifier, the SVM-KNN is a hybrid local and global classifier developed to mitigate the high variance typical of nearest neighbor methods and the curse-of-dimensionality. However, unlike the nearest means classifier of Mitani and Hamamoto, which is rooted in Euclidean space, the SVM-KNN can be used with any similarity function, as it assumes that the class information about the samples is captured by their pairwise

similarities without reference to the underlying feature space. Experiments on benchmark datasets using various similarity functions showed that SVM-KNN outperforms *k*-NN and its variants especially for cases with small training sets and large number of classes. SVM-KNN differs from local SDA because it is not a generative classifier.

Finally, note that different definitions of neighborhood may be used with local SDA. One could use the Mitani and Hamamoto (Mitani & Hamamoto, 2006) definition described above, or radius-based definitions. For example, the neighborhood of a test sample x may be defined as all the samples that fall within a factor of $1+\alpha$ of its similarity to its most similar neighbor, and α is cross-validated. This work employs the traditional definition of neighborhood, as the k nearest neighbors.

3.1 Consistency of the local SDA classifier

Generative classifiers with a finite number of model parameters, such as QDA or SDA, will not asymptotically converge to the Bayes classifier due to the model bias. This section shows that, like *k*-NN, the local SDA classifier is consistent such that its expected classification error $E[L]$ converges to the Bayes error rate L^* under the usual asymptotic assumptions that the number of training samples $N \rightarrow \infty$, the neighborhood size $k \rightarrow \infty$, but that the neighborhood size grows relatively slowly such that $k/N \rightarrow 0$. First a lemma is proven that will be used in the proof of the local SDA consistency theorem. Also, the known result that *k*-NN is a consistent classifier is reviewed in terms of similarity.

Let the similarity function be $s : \mathcal{B} \times \mathcal{B} \rightarrow \Omega$, where $\Omega \subset \mathbb{R}$ is discrete and let the largest element of Ω be termed s_{max}. Let X be a test sample and let the training samples $\{X_1, X_2, \ldots, X_N\}$ be drawn identically and independently. Re-order the training samples according to decreasing similarity and label them $\{Z_1, Z_2, \ldots, Z_N\}$ such that Z_k is the kth most similar neighbor of X.

Lemma 1 *Suppose* $s(x,Z) = s_{max}$ *if and only if* $x = Z$ *and* $P(s(x,Z) = s_{max}) > 0$ *where* Z *is a random training sample. Then* $P(s(x,Z_k) = s_{max}) \rightarrow 1$ *as* $k, N \rightarrow \infty$ *and* $k/N \rightarrow 0$.

Proof: The proof is by contradiction and is similar to the proof of Lemma 5.1 in (Devroye et al., 1996). Note that $s(x,Z_k) \neq s_{max}$ if and only if

$$\frac{1}{N}\sum_{i=1}^{N} I_{\{s(x,Z_i)=s_{max}\}} < \frac{k}{N}, \tag{22}$$

because if there are less than k training samples whose similarity to x is s_{max}, the similarity of the kth training sample to x cannot be s_{max}. The left-hand side of (22) converges to $P(s(x,Z) = s_{max})$ as $N \rightarrow \infty$ with probability one by the strong law of large numbers, and by assumption $P(s(x,Z) = s_{max}) > 0$. However, the right-hand side of (22) converges to 0 by assumption. Thus, assuming $s(x,Z_k) \neq s_{max}$ leads to a contradiction in the limit. Therefore, it must be that $s(x,Z_k) = s_{max}$.

Theorem 1 *Assume the conditions of Lemma 1. Define* L *to be the probability of error for test sample* X *given the training sample and label pairs* $\{(Z_1, Y_1), (Z_2, Y_2), \ldots, (Z_N, Y_N)\}$, *and let* L^* *be the Bayes error. If* $k, N \rightarrow \infty$ *and* $k/N \rightarrow 0$, *then for the local SDA classifier* $E[L] \rightarrow L^*$.

Proof: By Lemma 1, $s(x,Z_i) = s_{max}$ for $i \leq k$ in the limit as $N \rightarrow \infty$, and thus in the limit the centroid μ_h of the subset of the k neighbors that are from class h must satisfy $s(x, \mu_h) = s_{max}$, for every class h which is represented by at least one sample in the k neighbors. By definition

of the local SDA algorithm, any class $\bar{h}$ that does not have at least one sample in the k neighbors is assigned the class prior probability $P(Y = \bar{h}) = 0$, so it is effectively eliminated from the possible classification outcomes. Then, the constraint (9) on the expected value of the class-conditional similarity for every class g that is represented in the k neighbors of x is

$$E_{P(s(x,\mu_h)|Y=g)}[s(X,\mu_h)] = s_{max}, \tag{23}$$

which is solved by the pmf $P(s(x, \mu_h) \mid Y = g) = 1$ if $s(x, \mu_h) = s_{max}$, and zero otherwise. Thus the local SDA classifier (12) becomes

$$\hat{y} = \arg\max_{g=1,\ldots G} \hat{P}(Y = g), \tag{24}$$

where the estimated probability of each class $\hat{P}$ $(Y = g)$ is calculated using a maximum likelihood estimate of the class probabilities for the neighborhood. Then, $\hat{P}$ $(Y = g) \rightarrow P(Y = g \mid x)$ as $k \rightarrow \infty$ with probability one by the strong law of large numbers. Thus the local SDA classifier converges to the Bayes classifier, and the local SDA average error $E[L] \rightarrow L^*$.
The known result that k-NN is a consistent classifier can be stated in terms of similarity as a direct consequence of Lemma 1:
Lemma 2 *Assume the conditions of Lemma 1 and define L and L* as in Theorem 1. For the similarity-based k-NN classifier* $E[L] \rightarrow L^*$.
Proof. It follows directly from Lemma 1 that within the size-k neighborhood of x, $Z_i = x$ for $i \leq k$. Thus, the k-NN classifier (1) estimates the most frequent class among the k samples maximally similar to x:

$$\begin{aligned} \hat{y} &= \arg\max_{g=1,\ldots,G} \sum_{i=1}^{k} I(Y_i = g) \\ &= \hat{P}(Y = g). \end{aligned}$$

The summation converges to the class prior $P(Y = g \rightarrow x)$ as $k \rightarrow \infty$ with probability one by the strong law of large numbers, and the k-NN classifier becomes that in (24). Thus the similarity-based k-NN classifier is consistent.

4. Mixture SDA

Like LDA and QDA, basic SDA may be too biased if the similarity space - or more generally the descriptive statistics space - is multi-modal. In analogy to metric space mixture models, the bias problem in similarity space may be alleviated by generalizing the SDA formulation with similarity-based mixture models. In the *mixture SDA* models, the class-conditional probability distribution of the descriptive statistics $\mathcal{T}(x)$ for a test sample x is modeled as a weighted sum of exponential components. Generalizing the single centroid-based SDA classifier and drawing from the metric mixture models (Duda et al., 2001; Hastie et al., 2001), each class h is characterized by c_h centroids $\{\mu_{hl}\}$. The descriptive statistics for test sample x are its similarities to the centroids of class h, $\{s(x, \mu_{h1}), s(x, \mu_{h2}), \ldots, s(x, \mu_{hc_h})\}$, for each class h. The mixture SDA model for the probability of the similarities, assuming that test sample x is drawn from class g, is written as

$$P(s(x,\mu_{h1}), s(x,\mu_{h2}), \ldots, s(x,\mu_{hc_h})|Y=g) = \sum_{l=1}^{c_h} w_{ghl}\gamma_{ghl}e^{\lambda_{ghl}s(x,\mu_{hl})}, \quad (25)$$

where $\sum_{l=1}^{c_h} w_{ghl} = 1$ and $w_{ghl} > 0$. Then, the SDA classification rule (12) for mixture SDA becomes to classify x as the class $\hat{y}$ that solves the maximum a posteriori problem

$$\arg\max_{f=1,\ldots,G} \sum_{g=1}^{G} C(f,g) \left(\prod_{h=1}^{G} \sum_{l=1}^{c_h} w_{ghl}\gamma_{ghl}e^{\lambda_{ghl}s(x,\mu_{hl})} \right) P(Y=g). \quad (26)$$

Note how the mixture SDA generative model (25) parallels the metric mixture formulation of Gaussian mixture models (GMMs), with the exponentials $\gamma_{ghl}e^{\lambda_{ghl}s(x,\mu_{hl})}$ in place of the Gaussian components. However, there are deep differences between mixture SDA and metric mixture models. In metric learning, the mixtures model the underlying generative probability distributions of the features. Due to the curse of dimensionality, high-dimensional, multi-modal feature spaces require many training samples for robust model parameter estimation. For example, for d features, GMMs require that a $d \times 1$ mean vector and a $d \times d$ covariance matrix be estimated for each component in each class, for a total of $c_h \times (d^2+3d)/2$ parameters per mixture. Constraining each Gaussian covariance to be diagonal, at the cost of an increased number of mixture components, alleviates the robust estimation problem, but does not solve it (Reynolds & Rose, 1995).

When relatively few training samples are available, robust parameter estimation becomes particularly di±cult. In similarity-based learning the modeled quantity is the similarity of a sample to a class centroid. The estimation problem is essentially univariate and reduces to estimating the exponent λ_{ghl} in each component of the mixture, for a total of $c_h \times G \times 2$ parameters per mixture (the scaling parameter γ_{ghl} follows trivially). This simpler classifier architecture allows robust parameter estimation from smaller training set depending on the number of centroids per class, or, more generally, the number of descriptive statistics.

Another major difference between mixture SDA and metric mixture models is in the number of class-conditional probability models that must be estimated. In metric learning, G mixtures are estimated, one for each of the G possible classes from which a sample x may be drawn. In mixture SDA, G^2 mixture models are estimated. Each sample x is hypothesized drawn from class $g = 1, 2, \ldots G$, and its similarities to each of the G classes are modeled by the mixture (25), with $h = 1, 2, \ldots G$. When the number of classes grows, or when the number of components in each mixture model grows, the quadratic growth in the number of needed models presents a challenge in robust parameter estimation, especially when the number of available training samples is relatively small. However, this problem is mitigated by the fact that the component SDA parameters may be robustly estimated with smaller training sets than in metric mixture models due to the simpler, univariate estimation problem at the heart of SDA classification. The next section discusses the mixture SDA parameter estimation procedure.

4.1 Estimating the parameters for mixture SDA models

Computing the SDA mixture model for the similarities of samples $x \in \mathcal{X}_g$ to class h requires estimating the number of components c_h, the component centroids $\{\mu_{hl}\}$, the component

weights $\{w_{ghl}\}$ and the component SDA parameters $\{\lambda_{ghl}\}$ and $\{\gamma_{ghl}\}$. This section describes an EM algorithm for estimating these mixture parameters. The algorithm parallels the EM approach for estimating GMM parameters (Duda et al., 2001; Hastie et al., 2001); it is first summarized below, and then explained in detail in the following sections.

Let $\theta_{gh} = \{\{w_{ghl}\}, \{\gamma_{ghl}\}, \{\lambda_{ghl}\}\}$ for $l = 1, 2 \ldots c_h$ be the set of parameters for the class h mixture model to be estimated under the assumption that the training samples z_i, for $i = 1, 2, \ldots n_g$ are drawn identically and independently. Denote by C a random component of the mixture and by $P(C = l \mid s(z_i, \mu_{hl}), \theta_{gh})$ the responsibility (Hastie et al., 2001) of the lth component for the ith training sample similarity $s(z_i, \mu_{hl})$. Also write $P(s(z_i, \mu_{hl}) \mid C = l, \theta_{gh}) = \gamma_{ghl} e^{\lambda_{ghl} s(z_i, \mu_{hl})}$.

The proposed EM algorithm for mixture SDA is:

1. Compute the centroids $\{\mu_{hl}\}$ with K-medoids algorithm.
2. Initialize the parameters $\{w_{ghl}\}$ and the components $P(s(z_i, \mu_{hl}) \mid C = l, \theta_{gh})$.
3. E step: compute the responsibilities

$$P(C = l|s(z_i, \mu_{hl}), \theta_{gh}) = \frac{w_{ghl} P(s(z_i, \mu_{hl})|C = l, \theta_{gh})}{\sum_{l=1}^{c_h} w_{ghl} P(s(z_i, \mu_{hl})|C = l, \theta_{gh})}. \tag{27}$$

4. M step: compute model parameters
   (a) Find the λ_{ghl} which solves

$$E_{P(T(x)|Y=g)}[s(X, \mu_{hl})] = \frac{\sum_{i=1}^{n_g} s(z_i, \mu_{hl}) P(C = l|s(z_i, \mu_{hl}), \theta_{gh})}{\sum_{i=1}^{n_g} P(C = l|s(z_i, \mu_{hl}), \theta_{gh})}. \tag{28}$$

   (b) Compute the corresponding scaling factor

$$\gamma_{ghl} = \frac{1}{\sum\limits_{s(X,\mu_{hl}) \in \Omega} e^{\lambda_{ghl} s(X, \mu_{hl})}}. \tag{29}$$

   (c) Compute the component weights

$$w_{ghl} = \frac{1}{n_g} \sum_{i=1}^{n_g} P(C = l|s(z_i, \mu_{hl}), \theta_{gh}). \tag{30}$$

5. Repeat E and M steps until convergence criterion is satisfied.

Note that, just like EM for GMMs, the EM algorithm for mixture SDA involves iterating the E step, which estimates the responsibilities, and the M step, which estimates the parameters that maximize the expected log-likelihood of the training data. At each iteration of the M step, the explicit expression (30) updates the component weights. However, unlike EM for GMMs, the update expression for the component parameters (28) is implicit and must be solved numerically. Another difference between the GMM and SDA EM algorithms is in how the centroids are estimated. For GMMs, the component means $\{\mu_{hl}\}$, which are the metric centroids, are updated at each iteration of the M step. For mixture SDA, the centroids $\{\mu_{hl}\}$ are estimated at the beginning of the algorithm and kept constant throughout the iterations.

The update expressions for the mixture SDA parameters are derived from the expression of the expected log-likelihood of the observed similarities. A standard assumption in EM is that the observed data are independent and identically distributed given the class and mixture component. For mixture SDA, this assumption means that the training sample similarities $\{\mathcal{T}_g(z_i)\} = \{s(z_i, \mu_{hl})\}$, $z_i \in \mathcal{X}_g$ to the component centroids are identically distributed and conditionally independent given the lth class component. Then, the expected log-likelihood of $\{\mathcal{T}_g(z_i)\}$ is

$$L(\{\mathcal{T}_g(z_i)\}|\theta_{gh}) = \sum_{i=1}^{n_g}\sum_{h=1}^{G}\sum_{l=1}^{c_h} \log\left(w_{ghl}\gamma_{ghl}e^{\lambda_{ghl}s(z_i,\mu_{hl})}\right) P(C = l|s(z_i, \mu_{hl}), \theta_{gh}). \quad (31)$$

Using the properties of the logarithm and rearranging the terms, $L(\{\mathcal{T}_g(z_i)\} \mid \theta_{gh})$ splits into the terms depending on w_{ghl} and the terms depending on λ_{ghl} and γ_{ghl}:

$$\begin{aligned} L(\{\mathcal{T}_g(z_i)\}|\theta_{gh}) &= \sum_{i=1}^{n_g}\sum_{h=1}^{G}\sum_{l=1}^{c_h} \log(w_{ghl})P(C = l|s(z_i, \mu_{hl}), \theta_{gh}) \\ &+ \sum_{i=1}^{n_g}\sum_{h=1}^{G}\sum_{l=1}^{c_h} \log(\gamma_{ghl})P(C = l|s(z_i, \mu_{hl}), \theta_{gh}) \\ &+ \sum_{i=1}^{n_g}\sum_{h=1}^{G}\sum_{l=1}^{c_h} \lambda_{ghl}s(z_i, \mu_{hl})P(C = l|s(z_i, \mu_{hl}), \theta_{gh}). \end{aligned} \quad (32)$$

The standard EM approach to maximizing (32) is to set its partial derivatives with respect to the parameters to zero and solve the resulting equations. This is the approach adopted here for estimating the mixture SDA parameters θ_{gh} for all g, h.

The derivation of the expression for the component weights $\{w_{ghl}\}$ follows directly from (32); both the derivation of and the final expression for the component weights are identical to the metric mixtures case. Section 4.1.1 re-derives the well-known expression for w_{ghl}.

Applying the EM approach, however, does not lead to explicit expressions for $\{\lambda_{ghl}\}$ and $\{\gamma_{ghl}\}$. Instead, it leads to many single-parameter constraint expressions for the mean similarities of the training data to the mixture component centroids. These expressions are solved with the same numerical solver used in the single-centroid SDA classifier.

4.1.1 Estimating the component weights

To compute the log-likelihood-maximizing weights w_{ghl}, one uses the standard technique of taking the derivative of the log-likelihood with respect to w_{ghl}, setting it to zero, and solving the resulting expression for w_{ghl}. The constraint $\sum_{l=1}^{c_h} w_{ghl} = 1$ is taken into account with the Lagrange multiplier η:

$$\frac{\partial}{\partial w_{ghl}}\left\{L(\{\mathcal{T}_g(z_i)\}|\theta_{gh}) + \eta\left(\sum_{l=1}^{c_h} w_{ghl} - 1\right)\right\} = \sum_{i=1}^{n_g}\frac{1}{w_{ghl}}P(C = l|s(z_i, \mu_{hl}), \theta_{gh}) + \eta = 0,$$

which gives the well-known expression for the component weights of a mixture model in terms of the responsibilities:

$$w_{ghl} = \frac{1}{n_g} \sum_{i=1}^{n_g} P(C = l | s(z_i, \mu_{hl}), \theta_{gh}). \tag{33}$$

4.1.2 Estimating γ_{ghl} and λ_{ghl}

The same approach used for estimating the component weights $\{w_{ghl}\}$ is adopted to estimate the SDA parameters $\{\gamma_{ghl}\}$ and $\{\lambda_{ghl}\}$: Find the likelihood-maximizing values of the parameters by setting the corresponding partial derivatives to zero and solving the resulting equations. First, since each γ_{ghl} is simply a scaling factor that ensures that each mixture component is a probability mass function, one rewrites

$$\gamma_{ghl} = \frac{1}{\sum_{s(X,\mu_{hl}) \in \Omega} e^{\lambda_{ghl} s(X,\mu_{hl})}}, \tag{34}$$

where $X \in \mathcal{X}_g$ is a random sample from class g, $s(X, \mu_{hl})$ is its corresponding random similarity to component centroid μ_{hl}, and Ω is the set of all possible similarity values. Substituting (34) into (32), setting the partial derivative of $L(\{\mathcal{T}_h(z_i)\} \mid \theta_{gh})$ with respect to λ_{ghl} to zero, and rearranging the terms gives

$$\frac{\sum_{s(X,\mu_{hl}) \in \Omega} s(X, \mu_{hl}) e^{\lambda_{ghl} s(X,\mu_{hl})}}{\sum_{s(X,\mu_{hl}) \in \Omega} e^{\lambda_{ghl} s(X,\mu_{hl})}} \sum_{i=1}^{n_g} P(C = l | s(z_i, \mu_{hl}), \theta_{gh}) = \sum_{i=1}^{n_g} s(z_i, \mu_{hl}) P(C = l | s(z_i, \mu_{hl}), \theta_{gh}). \tag{35}$$

The first term on the left side of (35) is simply the definition of the expected value of the similarity of samples in class g to the lth centroid of class h. Thus, one rewrites (35)

$$E_{P(\mathcal{T}(x)|Y=g)}[s(X, \mu_{hl})] = \frac{\sum_{i=1}^{n_g} s(z_i, \mu_{hl}) P(C = l | s(z_i, \mu_{hl}), \theta_{gh})}{\sum_{i=1}^{n_g} P(C = l | s(z_i, \mu_{hl}), \theta_{gh})}. \tag{36}$$

Expression (36) is an equality constraint on the expected value of the similarity of samples $z_i \in \mathcal{X}_g$ to the component centroids μ_{hl} of class h. This is the same type of constraint that must be solved in the mean-constrained, maximum entropy formulation of single-centroid SDA (9). In (9), the mean similarity of samples from class g to the single centroid of class h is constrained to be equal to the observed average similarity. Analogously, in (36), the mean similarity of the samples from class g to the lth centroid of class h is constrained to be equal to the weighted sum of the observed similarities, where each similarity is weighted by its normalized responsibility. To solve for λ_{ghl}, one uses the same numerical procedure used to solve (9) and described in Section 2.1. Thus, solving for all the $\{\lambda_{ghl}\}$ requires solving the $G \times \sum_{h=1}^{G} c_h$ expressions of (36).

It is not surprising that taking the EM approach to estimating λ_{ghl} has lead to the same expressions for the mean constraints in the maximum entropy approach to density estimation. It is known that maximum likelihood (ML) - the foundation for EM - and