A TRANSDUCTIVE EXTENSION OF MAXIMUM ENTROPY/ITERATIVE SCALING FOR DECISION AGGREGATION IN DISTRIBUTED CLASSIFICATION

David J. Miller, Yanxin Zhang, and George Kesidis

Department of Electrical Engineering, The Pennsylvania State University, University Park
djmiller@engr.psu.edu, yanxin@psu.edu, kesidis@engr.psu.edu

ABSTRACT

Many ensemble classification systems apply supervised learning to design a function for combining classifier decisions, which requires common labeled training samples across the classifier ensemble. Without such data, fixed rules (voting, Bayes rule) are usually applied. [1] alternatively proposed a *transductive constraint-based* learning strategy to learn how to fuse decisions even without labeled examples. There, decisions on test samples were chosen to satisfy constraints measured by each local classifier. There are two main limitations of that work. First, feasibility of the constraints was not guaranteed. Second, heuristic learning was applied. Here we overcome both problems via a transductive extension of *maximum entropy/improved iterative scaling* for aggregation in distributed classification. This method is shown to achieve improved decision accuracy over the earlier transductive approach on a number of UC Irvine data sets.

Index Terms— distributed ensemble classification, maximum entropy, constraint-based learning, transductive learning, iterative scaling

1. INTRODUCTION

Ensemble systems form ultimate decisions by aggregating hard or soft decisions made by individual classifiers. Common labeled training data is needed if one is to jointly design the local classifiers in a supervised fashion as in boosting and mixture of experts, or to learn the aggregation function which combines classifier decisions. Without common labeled training data, fixed rules such as voting and Bayes rule are often applied. Fundamental deficiencies of fixed rule methods are: 1) individual classifiers might assume incorrect class prior probabilities [1]; 2) statistical dependencies between individual classifiers are ignored or improperly accounted for.

[1] proposed a *transductive*, *constraint-based* (CB) method, where each local classifier contributes statistical constraints that the aggregation function must satisfy through the decisions it makes on test samples. CB effectively corrects inaccurate local class priors in making fused decisions, accounts for dependencies between classifiers, and does so without *any* communication between local classifiers. [1] used a heuristic gradient descent method, which chose the ensemble posterior pmfs on test samples to encode the constraint probabilities, measured by local classifiers $P_g^{(j)}[\hat{C}_j|C=c]$. The aggregation function forms estimates of these constraints $P_m[\hat{C}_j|C=c]$ to minimize the cross entropy function:

$$R=\sum_{j=1}^{M_e} D(P_m[\hat{C}_j|C=c]P_m[C=c]\,||\,P_g^{(j)}[\hat{C}_j|C=c]P_m[C=c]), \quad (1)$$

where M_e is the number of local classifiers and $P_m[C=c]$ is the new (estimated) class prior. [1] showed the derivation and full explanation of this method. Two limitations of the heuristic learning are: 1) it does not guarantee feasibility of the constraints because the local classifier training support (on which constraint probabilities are measured) and the test support (on which model estimates are learned) are different; 2) even when the constraints are feasible, there is a *set* of feasible solutions. The heuristic solution in [1] is not unique and does not ensure good test set accuracy.

In this paper we overcome these problems by proposing a transductive extension of *maximum entropy/improved iterative scaling* [3] for aggregation in distributed classification. Our first contribution is to augment the test support with training support to ensure the feasibility of constraints. Since the constraint probabilities are *measured* on training support, this augmentation *guarantees* constraint feasibility. To further improve classification performance, we also impose a constraint on mass allocation on the test support, seeking to make it as large as possible. In this way, we try to satisfy constraints as "transductively" as possible, *i.e.*, making the least possible use of the support augmentation.

Our second contribution is to propose a *transductive iterative scaling* (TIS) algorithm based on the *maximum entropy* (ME) principle which ensures uniqueness of the solution. Given measured constraints, the ME solution has been justified as the "least-biased" solution from a number of theoretical standpoints [2]. We have found this approach achieves greater accuracy than both the heuristic CB method [1] and fixed aggregation rules.

2. CONSTRAINT-BASED DISTRIBUTED CLASSIFICATION SYSTEM

As Fig. 1 shows, there are M_e local classifiers and N_c classes; the j-th local classifier is *designed* based on its own (separate) training set $\widetilde{\mathcal{X}}_j=\{(\underline{\widetilde{x}}_i^{(j)},\widetilde{c}_i^{(j)}), i=1,\ldots,N_j\}$, where $\underline{\widetilde{x}}_i^{(j)}$ and $\widetilde{c}_i^{(j)}$

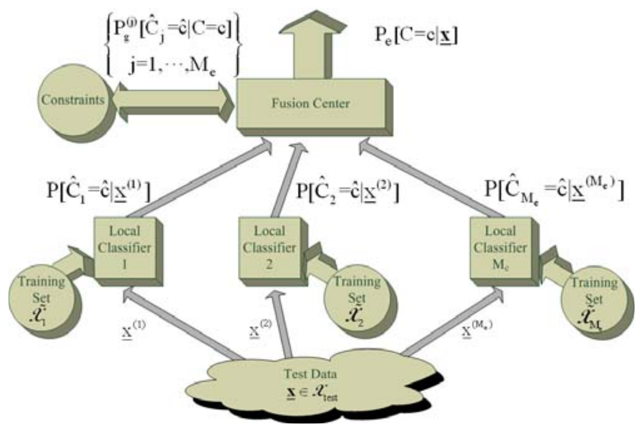

$\{P_g^{(j)}[\hat{C}_j=\hat{c}|C=c]$
$j=1,\cdots,M_e\}$
$P_e[C=c|\underline{x}]$
Constraints
Fusion Center
$P[\hat{C}_1=\hat{c}|\underline{x}^{(1)}]$
$P[\hat{C}_2=\hat{c}|\underline{x}^{(2)}]$
$P[\hat{C}_{M_e}=\hat{c}|\underline{x}^{(M_e)}]$
Local Classifier 1
Local Classifier 2
Local Classifier M_e
Training Set $\mathcal{X}_1$
Training Set $\mathcal{X}_2$
Training Set $\mathcal{X}_M$
$\underline{x}^{(1)}$
$\underline{x}^{(2)}$
$\underline{x}^{(M_e)}$
Test Data
$\underline{x} \in \mathcal{X}_{test}$

Fig. 1. Constraint-based distributed classification system.

are the feature vector and class label. We define $\mathcal{X}_j=\{\underline{\tilde{x}}_i^{(j)}\}$. A batch of test samples, $\mathcal{X}_{\text{test}}=\{\underline{\mathbf{x}}_1,\underline{\mathbf{x}}_2,\ldots,\underline{\mathbf{x}}_{N_{\text{test}}}\}$ are input to the distributed system with the concatenated vector $\underline{\mathbf{x}}=(\underline{x}^{(1)},\ldots,\underline{x}^{(M_e)})$. Here, $\underline{x}^{(j)}, j=1,\ldots,M_e$ is the component feature vector for classifier j. Classifier j produces hard/soft decisions $\{P_j[\hat{C}_j=\hat{c}|\underline{x}^{(j)}]\in[0,1], \hat{c}=1,\ldots,N_c, j=1,\ldots,M_e\}$ based on feature vector $\underline{x}^{(j)}$. The aggregation function makes final decisions $\{P_e[C=c|\underline{\mathbf{x}}], \underline{\mathbf{x}}\in\mathcal{X}_{\text{test}}\}$.

In our distributed setting, because there are no common labeled training data and there may be class prior probability mismatch between training and test data, [1] encoded conditional pairwise pmfs $\{P_g^{(j)}[\hat{C}_j=\hat{c}|C=c], \forall\hat{c}, c\}$ as the constraint probabilities from local classifier j. They are measured by

$$P_{\text{g}}^{(j)}[\hat{C}_j=\hat{c}|C=c]=\frac{\sum\limits_{i=1:\tilde{c}_i^{(j)}=c}^{N_j} P_j[\hat{C}_j=\hat{c}|\underline{\tilde{x}}_i^{(j)}]}{\sum\limits_{i=1:\tilde{c}_i^{(j)}=c}^{N_j} 1}, \forall j, c, \hat{c}. \qquad (2)$$

The aggregation function's transductive model *estimates* are

$$P_m[\hat{C}_j=\hat{c}|C=c]=\frac{\sum\limits_{i=1}^{N_{\text{test}}} P_e[C=c|\underline{\mathbf{x}}_i]P_j[\hat{C}_j=\hat{c}|\underline{x}_i^{(j)}]}{\sum\limits_{i=1}^{N_{\text{test}}} P_e[C=c|\underline{\mathbf{x}}_i]}, \forall j, c, \hat{c}. \quad (3)$$

The objective is to choose the ensemble posteriors $\{\{P_e[C=c|\underline{\mathbf{x}}_i]\forall c\}\}$ so that the transductive estimates match the constraints, *i.e.*,

$$P_m[\hat{C}_j=\hat{c}|C=c]\doteq P_g^{(j)}[\hat{C}_j=\hat{c}|C=c], \forall j, c, \hat{c}. \qquad (4)$$

This was heuristically achieved in [1] by minimizing (1).

3. ME/TIS METHOD

Supposing a uniform mass assignment to test support points, *i.e.*, $P_e[\underline{\mathbf{x}}]=\frac{1}{N_{\text{test}}}, \forall\underline{\mathbf{x}}\in\mathcal{X}_{\text{test}}$, the standard approach to finding a unique conditional distribution $\{P_e[c|\underline{\mathbf{x}}]\}$ satisfying given constraints (4) is to invoke the principle of *maximum entropy* [2]:

Parameters: $\{P[c|\underline{\mathbf{x}}], \underline{\mathbf{x}}\in\mathcal{X}_{\text{test}}\}$

Maximize:

$$H(C|\underline{\mathbf{X}})=-\sum_{\underline{\mathbf{x}}\in\mathcal{X}_{\text{test}}}\frac{1}{N_{\text{test}}}\sum_c P_e[c|\underline{\mathbf{x}}]\log P_e[c|\underline{\mathbf{x}}], \qquad (5)$$

Subject to:

$$\begin{aligned}&\sum_c P_e[c|\underline{\mathbf{x}}]=1 \quad \forall\underline{\mathbf{x}}\in\mathcal{X}_{\text{test}}\\ &P_m[\hat{C}_j=\hat{c}|C=c]=P_g^{(j)}[\hat{C}_j=\hat{c}|C=c] \quad \forall j, c, \hat{c}.\end{aligned} \qquad (6)$$

A serious difficulty with this approach is that the constraints may be infeasible because the constraints are *measured* using each local classifier's training support, but we are attempting to satisfy them using different (test) support. An example was given in [1]. To overcome this, we propose to *augment* the test support to *ensure* feasibility. The simplest way is to augment with local training support.

3.1. Augmentation with Local Training Supports

Since the constraints were measured on each local classifier's support, augmenting the test support with the local training sets guarantees constraint feasibility. Some care must be taken, however, to ensure that *sufficient* probability mass is allocated to the training supports to ensure constraint feasibility – *e.g.*, a *uniform* mass assignment to all support points, both test and training, will *not* in general ensure feasibility. Thus, we allow *flexible* allocation of probability mass to the training supports (both the total mass allocated to the training supports and how it is distributed across the training support points), choosing the joint pmf to have the form:

$$P_e[c,\{\underline{\mathbf{x}}\}]=\begin{cases}\frac{P_u}{N_{\text{test}}}P_e[c|\underline{\mathbf{x}}] & \underline{\mathbf{x}}\in\mathcal{X}_{\text{test}}\\ P[\underline{\tilde{x}}^{(j)},\tilde{c}^{(j)}] & \{\underline{\mathbf{x}}:\underline{x}^{(j)}=\underline{\tilde{x}}^{(j)}\}, (\underline{\tilde{x}}^{(j)},\tilde{c}^{(j)})\in\tilde{\mathcal{X}}_j\\ 0 & else.\end{cases} \qquad (7)$$

Here, the total mass allocated to $\mathcal{X}_{\text{test}}$ is $P_u=\sum\limits_{\underline{\mathbf{x}}\in\mathcal{X}_{\text{test}}}\sum\limits_c P_e[c,\underline{\mathbf{x}}]$ and each *test* sample is assigned equal mass $\frac{P_u}{N_{\text{test}}}$. P_u, $\{P_e[c|\underline{\mathbf{x}}]\}$ and $\{P[\underline{\tilde{x}}^{(j)},\tilde{c}^{(j)}]\}$, the probabilities assigned to the augmented support points, are all parameters we will learn.

Accordingly, we need compute the transductive estimates $P_m[\hat{C}_j=\hat{c}|C=c]=\frac{P_m[\hat{C}_j=\hat{c},C=c]}{P_m[C=c]}$, using the joint pmf (7). However, a difficulty is that (7) is defined on the support set $\mathcal{X}_{\text{test}}\bigcup_k\mathcal{X}_k$, but the posterior $P_j[\hat{C}_j=\hat{c}|\underline{x}^{(j)}]$ can *only* be evaluated on the support subset $\mathcal{X}_{\text{test}}\bigcup\mathcal{X}_j$. This means we cannot use full support to measure $P_m[\hat{C}_j=\hat{c}|C=c]$. We resolve this issue by *conditioning*. Let $\mathcal{X}_r^{(j)}=\{\{\underline{\mathbf{x}}\in\mathcal{X}_{\text{test}}\}\bigcup\{\underline{\mathbf{x}}:\underline{x}^{(j)}\in\mathcal{X}_j\}\}$, then we measure

$$P_m[\hat{C}_j=\hat{c}|C=c,\underline{\mathbf{x}}\in\mathcal{X}_r^{(j)}]=\frac{P_m[\hat{C}_j=\hat{c},C=c,\underline{\mathbf{x}}\in\mathcal{X}_r^{(j)}]}{P_m[C=c,\underline{\mathbf{x}}\in\mathcal{X}_r^{(j)}]}. \qquad (8)$$

Letting $N_m[\hat{C}_j=\hat{c},C=c,\underline{\mathbf{x}}\in\mathcal{X}_r^{(j)}]\equiv K_0P_m[\hat{C}_j=\hat{c},C=c,\underline{\mathbf{x}}\in\mathcal{X}_r^{(j)}]$ and $N_m[C=c,\underline{\mathbf{x}}\in\mathcal{X}_r^{(j)}]\equiv K_0P_m[C=c,\underline{\mathbf{x}}\in\mathcal{X}_r^{(j)}]$, where K_0 is the same normalization constant in both equations, ensuring these pmfs both sum to 1, we have

$$P_m[\hat{C}_j=\hat{c}|C=c,\underline{\mathbf{x}}\in\mathcal{X}_r^{(j)}]=\frac{N_m[\hat{C}_j=\hat{c},C=c,\underline{\mathbf{x}}\in\mathcal{X}_r^{(j)}]}{N_m[C=c,\underline{\mathbf{x}}\in\mathcal{X}_r^{(j)}]}. \qquad (9)$$

The notation $N_m[\cdot]$ reflects the fact that this quantity represents the expected number of occurrences of the joint event for $\underline{\mathbf{x}} \in \mathcal{X}_r^{(j)}$. Here,

$$N_m[C{=}c, \underline{\mathbf{x}}{\in}\mathcal{X}_r^{(j)}]{=}\frac{P_u}{N_{\text{test}}}\sum_{\underline{\mathbf{x}}\in\mathcal{X}_{\text{test}}}P_e[c|\underline{\mathbf{x}}] + \sum_{\underline{\widetilde{x}}^{(j)}\in\mathcal{X}_j,\widetilde{c}^{(j)}=c}P[\underline{\widetilde{x}}^{(j)},\widetilde{c}^{(j)}], \tag{10}$$

$$N_m[\hat{C}_j{=}\hat{c}, C{=}c, \underline{\mathbf{x}}{\in}\mathcal{X}_r^{(j)}]{=}\frac{P_u}{N_{\text{test}}}\sum_{\underline{\mathbf{x}}\in\mathcal{X}_{\text{test}}}P_j[\hat{C}_j{=}\hat{c}|\underline{x}^{(j)}]P_e[c|\underline{\mathbf{x}}] + \sum_{\underline{\widetilde{x}}^{(j)}\in\mathcal{X}_j,\widetilde{c}^{(j)}=c}P[\hat{C}_j{=}\hat{c}|\underline{\widetilde{x}}^{(j)}]P_j[\underline{\widetilde{x}}^{(j)},\widetilde{c}^{(j)}]. \tag{11}$$

From (9)-(11), the constraints (4) are nonlinear in the parameters that need to be learned, but IIS algorithm [3] requires linear constraints. Whereas, it is possible to *relax* the constraints (4), based on (9), to linear ones. In particular, *assuming* $N_m[C{=}c, \underline{\mathbf{x}}{\in}\mathcal{X}_r^{(j)}]{>}0$, we may multiply (4) through by $N_m[C{=}c, \underline{\mathbf{x}}{\in}\mathcal{X}_r^{(j)}]$ and write the equivalent linear constraints:

$$N_m[\hat{C}_j{=}\hat{c}, C{=}c, \underline{\mathbf{x}}{\in}\mathcal{X}_r^{(j)}]{=}P_g^{(j)}[\hat{C}_j{=}\hat{c}|C{=}c]N_m[C{=}c, \underline{\mathbf{x}}{\in}\mathcal{X}_r^{(j)}]. \tag{12}$$

To make test set posteriors as accurate as possible, they should contribute as much as possible to constraint satisfaction, *i.e.*, we have the following loosely stated learning principle: seek the *minimal* use of the extra training support necessary to achieve the constraints. We add an extra constraint $1-P_u = P_o^*$ and seek to find the *minimum* value P_o^* such that the constraints are still feasible. When the test set support is sufficient to meet the constraints, $P_o^* = 0$; otherwise $P_o^* > 0$.

Finally, we define the transductive ME distributed problem as:

Parameters: $P_u, \{P[c|\underline{\mathbf{x}}], \underline{\mathbf{x}}{\in}\mathcal{X}_{\text{test}}\}, \{p[\underline{\widetilde{x}}^{(j)},\widetilde{c}^{(j)}], (\underline{\widetilde{x}}^{(j)},\widetilde{c}^{(j)}){\in}\widetilde{\mathcal{X}}_j\}$

Maximize:

$$H(C,\underline{\mathbf{X}}) = -\sum_{\underline{\mathbf{x}}\in\mathcal{X}_{\text{test}}}\frac{P_u}{N_{test}}\sum_c P_e[c|\underline{\mathbf{x}}]\log\left(\frac{P_u}{N_{\text{test}}}P_e[c|\underline{\mathbf{x}}]\right) - \sum_j\sum_{(\underline{\widetilde{x}}^{(j)},\widetilde{c}^{(j)})\in\widetilde{\mathcal{X}}_j}P[\underline{\widetilde{x}}^{(j)},\widetilde{c}^{(j)}]\log P[\underline{\widetilde{x}}^{(j)},\widetilde{c}^{(j)}] \tag{13}$$

Subject to:

$$N_m[\hat{C}_j{=}\hat{c}, C{=}c, \underline{\mathbf{x}}{\in}\mathcal{X}_r^{(j)}]{=}P_g^{(j)}[\hat{C}_j{=}\hat{c}|C{=}c]N_m[C{=}c, \underline{\mathbf{x}}{\in}\mathcal{X}_r^{(j)}] \quad \forall j,c,\hat{c}$$
$$\sum_c P_e[C=c|\underline{\mathbf{x}}] = 1 \quad \forall\underline{\mathbf{x}}\in\mathcal{X}_{\text{test}} \tag{14}$$
$$P_u + \sum_j\sum_{(\underline{\widetilde{x}}^{(j)},\widetilde{c}^{(j)})\in\widetilde{\mathcal{X}}_j}P[\underline{\widetilde{x}}^{(j)},\widetilde{c}^{(j)})] = 1$$
$$1-P_u = P_o^*$$

3.2. Transductive Iterative Scaling (TIS) Algorithm

We invoke the method of Lagrange Multipliers to solve (13)-(14). $\{\gamma(\hat{C}_j{=}\hat{c}, C{=}c)\forall j,c,\hat{c}\}$ are the Lagrange Multipliers associated with the local classifier constraints, which need to be learned. α and $\lambda(\underline{\mathbf{x}}), \forall\underline{\mathbf{x}}{\in}\mathcal{X}_{\text{test}}$ will be automatically chosen to satisfy the sum constraints. β is an "external" parameter used to set $1-P_u$ as previously discussed. The Lagrangian cost function is

$$\begin{aligned}L(\beta) = &-\sum_{\underline{\mathbf{x}}\in\mathcal{X}_{\text{test}}}\frac{P_u}{N_{\text{test}}}\sum_c P_e[c|\underline{\mathbf{x}}]\log P_e[c|\underline{\mathbf{x}}] - P_u\log\frac{P_u}{N_{\text{test}}}\\
&-\sum_j\sum_{(\underline{\widetilde{x}}^{(j)},\widetilde{c}^{(j)})\in\widetilde{\mathcal{X}}_j}P[\underline{\widetilde{x}}^{(j)},\widetilde{c}^{(j)}]\log P[\underline{\widetilde{x}}^{(j)},\widetilde{c}^{(j)}]\\
&+\sum_{j,c,\hat{c}}\gamma(\hat{C}_j{=}\hat{c}, C{=}c)(N_m[\hat{C}_j{=}\hat{c}, C{=}c, \underline{\mathbf{x}}{\in}\mathcal{X}_r^{(j)}] -\\
&\quad P_g^{(j)}[\hat{C}_j = \hat{c}|C=c]N_m[C=c, \underline{\mathbf{x}}\in\mathcal{X}_r^{(j)}])\\
&+\alpha(P_u + \sum_j\sum_{(\underline{\widetilde{x}}^{(j)},\widetilde{c}^{(j)})\in\widetilde{\mathcal{X}}_j}P[\underline{\widetilde{x}}^{(j)},\widetilde{c}^{(j)}] - 1)\\
&+\sum_{\underline{\mathbf{x}}\in\mathcal{X}_{\text{test}}}\lambda(\underline{\mathbf{x}})(\sum_c P[c|\underline{\mathbf{x}}] - 1) + \beta(1-P_u)\end{aligned} \tag{15}$$

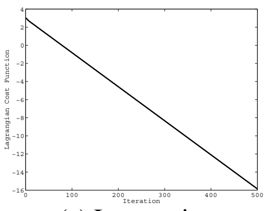
The TIS algorithm consists of alternating i) optimization of P_u, $\{P_e[c|\underline{\mathbf{x}}]\}$ and $\{P[\underline{\widetilde{x}}^{(j)},\widetilde{c}^{(j)}]\}$ given $\{\gamma\}$ held fixed, followed by ii) update of $\{\gamma\}$ given the other parameters fixed. At fixed β, in each iteration we measure the *Deviation* D, *i.e.*, the squared Euclidean distance between $N_m[\hat{C}_j{=}\hat{c}, C{=}c, \underline{\mathbf{x}}{\in}\mathcal{X}_r^{(j)}]$ and $P_g^{(j)}[\hat{C}_j{=}\hat{c}|C{=}c]N_m[C{=}c, \underline{\mathbf{x}}{\in}\mathcal{X}_r^{(j)}]$, summed over all $j,c,\hat{c}$, and stop when ΔD is less than a threshold value. TIS algorithm solves a convex optimization problem, descending in $L(\beta)$ and converging to the unique ME solution at each β for which the constraints are feasible. If the problem is infeasible at a given β, convergence of the algorithm is not guaranteed. When $\beta{\to}-\infty$, $P_u{=}1$ and we are seeking a solution that *only* relies on the test set support. When $\beta{\to}\infty$, $P_u{=}0$ and we are not using the test support at all. When $\beta{=}0$, there is *no* constraint on P_u – this solution thus achieves highest entropy $H(C,\underline{\mathbf{X}})$, compared to solutions at other β values.

Furthermore, we use an external bisection search over β for P_o^*. The *overall* algorithm terminates when one of two conditions is satisfied: 1) At the current β, constraint satisfaction is not achieved. This indicates that P_o^* is greater than the value $1-P_u$ associated with the current β. 2) At the current β, at termination of TIS, $1-P_u < \delta$, where δ is a small number. In this case, we have essentially found that there is an ME solution satisfying the constraints using only the test support, *i.e.*, $P_o^* \approx 0$.

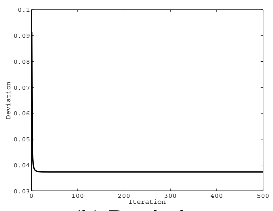
4. EXPERIMENTS

[1] gave an example of constraint nonsatisfiability, *i.e.*, the heuristic CB method in [1] could not satisfy all constraints *only* using $\{\{P_e[c|\underline{\mathbf{x}}]\}, \underline{\mathbf{x}}\in\mathcal{X}_{\text{test}}\}$. This is still true for the ME/TIS method if no support augmentation is used. Fig. 2 shows the Lagrangian cost function and Deviation for ME/TIS with and without training support augmentation. The upper two figures show the Lagrangian monotonically decreases without convergence and the Deviation does not approach zero, which means the solution is infeasible. The lower two figures show ME/TIS ($\beta{=}0$) with local training support augmentation and demonstrate a feasible solution is achieved.

Next, we evaluated on data sets from the UC Irvine machine learning repository that were also used in [1] and we

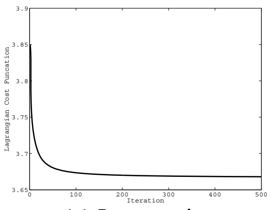


(a) Lagrangian

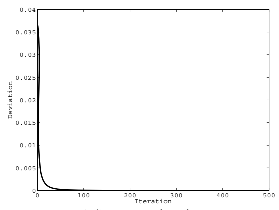


(b) Deviation

ME/TIS without training support augmentation



(c) Lagrangian



(d) Deviation

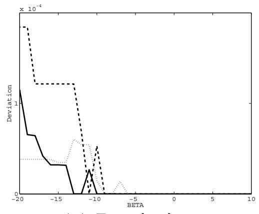
ME/TIS with training support augmentation

Fig. 2. Constraint satisfiability for ME/TIS algorithm.

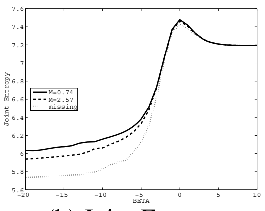
followed the experimental protocol from [1]. To simulate a distributed classification environment we used five local classifiers, each a naive Bayes classifier working on a randomly selected subset of features. All features in the data sets are continuous-valued and were modeled by (class-conditional) Gaussian densities. We performed five replications of two-fold cross-validation for all data sets and measured the average error rate over all ten test folds. We evaluated classification accuracy for varying test set priors. A new test set with given priors was obtained by sampling with replacement from the original test set. We quantified the mismatch between test set priors ($P_{\text{test}}[C=c]$) and local training set priors ($P^{(j)}_{\text{trn}}[C=c]$) by the sum of cross entropies:

$$M=\sum_{j=1}^{M_e}\sum_{c=1}^{N_c}P^{(j)}_{\text{trn}}[C=c]\log\left(\frac{P^{(j)}_{\text{trn}}[C=c]}{P_{\text{test}}[C=c]}\right). \qquad (16)$$

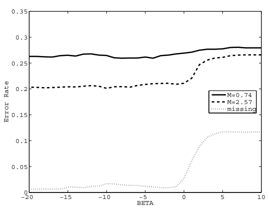
We also considered the case where one class was entirely missing from the test set.



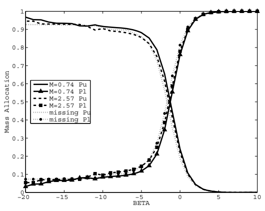
(a) Deviation



(b) Joint Entropy



(c) Classification Error Rate



(d) Mass Allocation

Fig. 3. The influence of β on TIS algorithm (diabetes).

Fig. 3 shows the influence of β on the ME/TIS for *diabetes*. We measured four different values: Deviation, joint entropy, classification error rate on the test data, and mass allocation (P_u on the test support and P_l=1–P_u on the training support). We plotted curves for 3 mismatch cases. We changed β from -20 to 10 with $\Delta\beta = 1$ and, for each β, we ran ME/TIS until $\Delta D < 10^{-9}$. In (a), when $\beta < -6$, the Deviation does not approach 0 when ME/TIS terminates. This demonstrates constraint infeasibility may occur at finite β (when P_l is made too small). ME/TIS achieves peak entropy at $\beta = 0$ as (b) shows (no constraint on mass allocation) and when β goes either negative or positive, the joint entropy decreases. In (d), as β becomes more negative, the test support gets more probability mass allocation P_u. However, the rate of P_u increase decreases as β becomes more negative, because it is difficult to satisfy the local constraints if the TIS algorithm assigns too much probability mass to the test set.

data set	M	ME/TIS		heuristic CB	
		Err	Ent	Err	Ent
vehicle	0.98	0.53	0.86	0.59	0.08
	2.73	0.49	0.76	0.60	0.11
	missing	0.47	0.73	0.62	0.09
diabetes	0.74	0.26	0.43	0.28	0.03
	2.57	0.21	0.30	0.27	0.01
	missing	0.01	0.11	0.18	0.02
sonar	0.89	0.28	0.33	0.32	10^{-3}
	2.28	0.20	0.25	0.33	10^{-6}
	missing	0.16	0.21	0.34	10^{-6}

Table 1. Classification performance (error rate and conditional entropy on the test set) comparison between ME/TIS and the heuristic CB method in [1] .

In Table 1, we compare ME/TIS with the heuristic CB method in [1]. We can see ME/TIS achieves better classification error rates than the heuristic CB method and higher entropy as we would expect, since it seeks the maximum entropy solution.

5. CONCLUSION

In this work, we have proposed a new ME framework for transductive learning of decision aggregation rules when there is no common labeled data across local classifiers. The new approach overcomes constraint infeasibility and nonuniqueness of the solution and achieves better results than the heuristic CB method in [1].

6. REFERENCES

[1] D.J. Miller and S. Pal, "Transductive methods for distributed ensemble classification," *Neural Computation*, vol. 19, pp. 856–884, March 2007.

[2] E.T. Jaynes, in *Papers on probability, statistics, and statistical physics*. Kluwer Academic Publishers, 1989.

[3] S. Della Pietra, A.L. Berger and V.J. Della Pietra, "A maximum entropy approach to natural language processing," *Computational Linguistics*, vol.22, pp.39–71, 1996.