Cross-Instance Tuning of Unsupervised Document Clustering Algorithms*

Damianos Karakos, Jason Eisner and Sanjeev Khudanpur
Center for Language and Speech Processing
Johns Hopkins University
Baltimore, MD 21218
{damianos,eisner,khudanpur}@jhu.edu

Carey E. Priebe
Dept. of Applied Mathematics and Statistics
Johns Hopkins University
Baltimore, MD 21218
cep@jhu.edu

Abstract

In unsupervised learning, where no training takes place, one simply hopes that the unsupervised learner will work well on *any* unlabeled test collection. However, when the variability in the data is large, such hope may be unrealistic; a *tuning* of the unsupervised algorithm may then be necessary in order to perform well on new test collections. In this paper, we show how to perform such a tuning in the context of unsupervised document clustering, by (i) introducing a degree of freedom, α, into two leading information-theoretic clustering algorithms, through the use of generalized mutual information quantities; and (ii) selecting the value of α based on clusterings of similar, but *supervised* document collections (cross-instance tuning). One option is to perform a tuning that directly minimizes the error on the supervised data sets; another option is to use "strapping" (Eisner and Karakos, 2005), which builds a classifier that learns to distinguish good from bad clusterings, and then selects the α with the best predicted clustering on the test set. Experiments from the "20 Newsgroups" corpus show that, although both techniques improve the performance of the baseline algorithms, "strapping" is clearly a better choice for cross-instance tuning.

*This work was partially supported by the DARPA GALE program (Contract No. HR0011-06-2-0001) and by the JHU WSE/APL Partnership Fund.

1 Introduction

The problem of combining labeled and unlabeled examples in a learning task *(semi-supervised learning)* has been studied in the literature under various guises. A variety of algorithms (e.g., bootstrapping (Yarowsky, 1995), co-training (Blum and Mitchell, 1998), alternating structure optimization (Ando and Zhang, 2005), etc.) have been developed in order to improve the performance of supervised algorithms, by automatically extracting knowledge from lots of *unlabeled* examples. Of special interest is the work of Ando and Zhang (2005), where the goal is to build many supervised auxiliary tasks from the unsupervised data, by creating artificial labels; this procedure helps learn a transformation of the input space that captures the relatedness of the auxiliary problems to the task at hand. In essence, Ando and Zhang (2005) transform the semi-supervised learning problem to a *multi-task learning* problem; in multi-task learning, a (usually large) set of *supervised* tasks is available for training, and the goal is to build models which can *simultaneously* do well on all of them (Caruana, 1997; Ben-David and Schuller, 2003; Evgeniou and Pontil, 2004).

Little work, however, has been devoted to study the situation where lots of labeled examples, of one kind, are used to build a model which is tested on unlabeled data of a "different" kind. This problem, which is the topic of this paper, cannot be cast as a multi-task learning problem (since there are labeled examples of only one kind), neither can be cast as a semi-supervised problem (since there are no training labels for the test task). Note that we are interested in the case where the hidden test labels may have no semantic relationship with the training labels; in

Proceedings of NAACL HLT 2007, pages 252–259,
Rochester, NY, April 2007.

some cases, there may not even be any information about the test labels—what they represent, how many they are, or at what granularity they describe the data. This situation can arise in the case of unsupervised clustering of documents from a large and diverse corpus: it may not be known in what way the resulting clusters split the corpus (is it in terms of topic? genre? style? authorship? a combination of the above?), unless one inspects each resulting cluster to determine its "meaning."

At this point, we would like to differentiate between two concepts: a target *task* refers to a class of problems that have a common, high-level description (e.g., the text document clustering task, the speech recognition task, etc.). On the other hand, a task *instance* refers to a particular example from the class. For instance, if the task is *"document clustering,"* a task instance could be *"clustering of a set of scientific documents into particular fields"*; or, if the task is *"parsing,"* a task instance could be *"parsing of English sentences from the Wall Street Journal corpus"*. For the purposes of this paper, we further assume that there are task instances which are *unrelated*, in the sense that that there are no common labels between them. For example, if the task is *"clustering from the 20 Newsgroups corpus,"* then *"clustering of the computer-related documents into PC-related and Mac-related"* and *"clustering of the politics-related documents into Middle-East-related and non-Middle-East-related"* are two distinct, unrelated instances. In more mathematical terms, if task instances T_1, T_2 take sets of observations $\boldsymbol{X}_1, \boldsymbol{X}_2$ as input, and try to predict labels from sets S_1, S_2, respectively, then they are called unrelated if $X_1 \cap X_2 = \varnothing$ and $S_1 \cap S_2 = \varnothing$.

The focus of this paper is to study the problem of *cross-instance tuning* of unsupervised algorithms: how one can tune an algorithm, which is used to solve a particular task instance, using knowledge from an unrelated task instance. To the best of our knowledge, this cross-instance learning problem has only been tackled in (Eisner and Karakos, 2005), whose "strapping" procedure learns a meta-classifier for distinguishing good from bad clusterings.

In this paper, we introduce a scalar parameter α (a new degree of freedom) into two basic unsupervised clustering algorithms. We can tune α to maximize unsupervised clustering performance on *different* task instances where the correct clustering is known. The hope is that tuning the parameter learns something about the task in general, which transfers from the supervised task instances to the unsupervised one. Alternatively, we can tune a meta-classifier so as to select good values of α on the supervised task instances, and then use the same meta-classifier to select a good (possibly different) value of α in the unsupervised case.

The paper is organized as follows: Section 2 gives a background on text categorization, and briefly describes the algorithms that we use in our experiments. Section 3 describes our parameterization of the clustering algorithms using Jensen-Rényi divergence and Csiszár's mutual information. Experimental results from the "20 Newsgroups" data set are shown in Section 4, along with two techniques for cross-instance learning: (i) "strapping," which, at test time, picks a parameter based on various "goodness" cues that were learned from the labeled data set, and (ii) learning the parameter from a supervised data set which is chosen to statistically match the test set. Finally, concluding remarks appear in Section 5.

2 Document Categorization

Document categorization is the task of deciding whether a piece of text belongs to any of a set of prespecified categories. It is a generic text processing task useful in indexing documents for later retrieval, as a stage in natural language processing systems, for content analysis, and in many other roles (Lewis and Hayes, 1994). Here, we deal with the unsupervised version of document categorization, in which we are interested in clustering together documents which (hopefully) belong to the same topic, without having any training examples.[1] *Supervised* information-theoretic clustering approaches (Torkkola, 2002; Dhillon et al., 2003) have been shown to be very effective, even with a small amount of labeled data, while *unsupervised* methods (which are of particular interest to us) have been shown to be competitive, matching the classification accuracy of supervised methods.

Our focus in this paper is on document categorization algorithms which use information-theoretic

[1] By this, we mean that training examples having the same category labels as the test examples are not available.

criteria, since there are natural ways of generalizing these criteria through the introduction of tunable parameters. We use two such algorithms in our experiments, the sequential Information Bottleneck (sIB) and Iterative Denoising Trees (IDTs); details about these algorithms appear below.

A note on mathematical notation: We assume that we have a collection $\mathcal{A} = \{X(1), \ldots, X(N)\}$ of N documents. Each document $X(i)$ is essentially a "bag of words", and induces an empirical distribution $\hat{P}_{X(i)}$ on the vocabulary $\mathcal{X}$. Given a subset (cluster) C of documents, the conditional distribution on $\mathcal{X}$, given the cluster, is just the centroid: $\hat{P}_{X|C} = \frac{1}{|C|}\sum_{X(i)\in C}\hat{P}_{X(i)}$. If a subcollection $S \subset \mathcal{A}$ of documents is partitioned into clusters $C_1, \ldots, C_m$, and each document $X(i) \in S$ is assigned to a cluster $C_{Z(i)}$, where $Z(i) \in \{1, \ldots, m\}$ is the cluster index, then the mutual information between words and corresponding clusters is given by

$$I(X;Z|S) = \sum_{z\in\{1,\ldots,m\}} P(z|S)D(\hat{P}_{X|C_z}\|\hat{P}_{X|S}),$$

where $P(z|S) \triangleq |C_z|/|S|$ is the "prior" distribution on the clusters and $D(\cdot\|\cdot)$ is the Kullback-Leibler divergence (Cover and Thomas, 1991).

2.1 The Information Bottleneck Method

The Information Bottleneck (IB) method (Tishby et al., 1999; Slonim and Tishby, 2000; Slonim et al., 2002) is one popular approach to unsupervised categorization. The goal of the IB (with "hard" clustering) is to find clusters such that the mutual information $I(X;Z)$ between words and clusters is as large as possible, under a constraint on the number of clusters. The procedure for finding the maximizing clustering in (Slonim and Tishby, 2000) is agglomerative clustering, while in (Slonim et al., 2002) it is based on many random clusterings, combined with a sequential update algorithm, similar to K-means. The update algorithm re-assigns each data point (document) d to its most "similar" cluster C, in order to *minimize* $I(X;Z|C\cup\{d\})$, i.e.,

$$\delta D(\hat{P}_{X|\{d\}}\|\hat{P}_{X|\{d\}\cup C}) + (1-\delta)D(\hat{P}_{X|C}\|\hat{P}_{X|\{d\}\cup C}),$$

where $\delta = \frac{1}{|C|+1}$. This latter procedure is called the *sequential Information Bottleneck* (sIB) method, and is considered the state-of-the-art in unsupervised document categorization.

2.2 Iterative Denoising Trees

Decision trees are a powerful technique for equivalence classification, accomplished through a recursive successive refinement (Jelinek, 1997). In the context of unsupervised classification, the goal of decision trees is to cluster empirical distributions (bags of words) into a given number of classes, with each class corresponding to a leaf in the tree. They are built top-down (as opposed to the bottom-up construction in IB) using maximization of mutual information between words and clusters $I(X;Z|t)$ to drive the splitting of each node t; the hope is that each leaf will contain data points which belong to only one latent category.

Iterative Denoising Trees (also called Integrated Sensing and Processing Decision Trees) were introduced in (Priebe et al., 2004a), as an extension of regular decision trees. Their main feature is that they *transform* the data at each node, before splitting, by projecting into a low-dimensional space. This transformation corresponds to feature extraction; different features are suppressed (or amplified) by each transformation, depending on what data points fall into each node (*corpus-dependent-feature-extraction* property (Priebe et al., 2004b)). Thus, dimensionality reduction and clustering are chosen so that they *jointly* optimize the local objective.

In (Karakos et al., 2005), IDTs were used for an unsupervised hyperspectral image segmentation application. The objective at each node t was to maximize the mutual information between spectral components and clusters given the pixels at node t, computed from the *projected* empirical distributions. At each step of the tree-growing procedure, the node which yielded the highest increase in the average, per-node mutual information, was selected for splitting (until a desired number of leaves was reached). In (Karakos et al., 2007b), the mutual information objective was replaced with a parameterized form of mutual information, namely the Jensen-Rényi divergence (Hero et al., 2001; Hamza and Krim, 2003), of which more details are provided in the next section.

3 Parameterizing Unsupervised Clustering

As mentioned above, the algorithms considered in this paper (sIB and IDTs) are unsupervised, in the

sense that they can be applied to test data without any need for tuning. Our procedure of adapting them, based on some supervision on a different task instance, is by introducing a parameter into the unsupervised algorithm. At least for simple cross-instance tuning, this parameter represents the information which is passed between the supervised and the unsupervised instances.

The parameterizations that we focused on have to do with the information-theoretic *objectives* in the two unsupervised algorithms. Specifically, following (Karakos et al., 2007b), we replace the mutual information quantities in IDTs as well as sIB with the *parameterized* mutual information measures mentioned above. These two quantities provide estimates of the dependence between the random quantities in their arguments, just as the usual mutual information does, but also have a scalar parameter $\alpha \in (0, 1]$ that controls the sensitivity of the computed dependence on the details of the joint distribution of X and Z. As a result, the effect of data sparseness on estimation of the joint distribution can be mitigated when computing these measures.

3.1 Jensen-Rényi Divergence

The Jensen-Rényi divergence was used in (Hero et al., 2001; Hamza and Krim, 2003) as a measure of similarity for image classification and retrieval. For two discrete random variables X, Z with distributions P_X, P_Z and conditional $P_{X|Z}$, it is defined as

$$I_\alpha(X;Z) = H_\alpha(P_X) - \sum_z P_Z(z) H_\alpha(P_{X|Z}(\cdot|z)), \tag{1}$$

where $H_\alpha(\cdot)$ is the Rényi entropy, given by

$$H_\alpha(P) = \frac{1}{1-\alpha} \log \left(\sum_{x \in \mathcal{X}} P(x)^\alpha \right), \quad \alpha \neq 1. \tag{2}$$

If $\alpha \in (0, 1)$, H_α is a concave function, and hence $I_\alpha(X;Z)$ is non-negative (and it is equal to zero if and only if X and Z are independent). In the limit as $\alpha \to 1$, $H_\alpha(\cdot)$ approaches the Shannon entropy (not an obvious fact), so $I_\alpha(\cdot)$ reduces to the regular mutual information. Similarly, we define

$$I_\alpha(X;Z|W) = \sum_w P_W(w) I_\alpha(X;Z|W=w),$$

where $I_\alpha(X;Z|W=w)$ is computed via (1) using the conditional distribution of X and Z given W.

Except in trivial cases, $H_\alpha(\cdot)$ is *strictly larger* than $H(\cdot)$ when $0 < \alpha < 1$; this means that the effects of extreme sparsity (few words per document, or too few occurrences of non-frequent words) on the estimation of entropy and mutual information can be dampened with an appropriate choice of α. This happens because extreme sparsity in the data yields empirical distributions which lie at, or close to, the boundary of the probability simplex. The entropy of such distributions is usually underestimated, compared to the smooth distributions which generate the data. Rényi's entropy is larger than Shannon's entropy, especially in those regions close to the boundary, and can thus provide an estimate which is closer to the true entropy.

3.2 Csiszár's Mutual Information

Csiszár defined the mutual information of order α as

$$I^C_\alpha(X;Z) = \min_Q \sum_z P_Z(z) D_\alpha(P_{X|Z}(\cdot|z) \| Q(\cdot)), \tag{3}$$

where $D_\alpha(\cdot\|\cdot)$ is the Rényi divergence (Csiszár, 1995). It was shown that $I^C_\alpha(X;Z)$ retains most of the properties of $I(X;Z)$—it is a non-negative, continuous, and concave function of P_X, it is convex in $P_{X|Z}$ for $\alpha < 1$, and converges to $I(X;Z)$ as $\alpha \to 1$.

Notably, $I^C_\alpha(X;Z) \leq I(X;Z)$ for $0 < \alpha < 1$; this means, as above, that α regulates the overestimation of mutual information that may result from data sparseness.

There is no analytic form for the minimizer of the right-hand-side of (3) (Csiszár, 1995), but it may be computed via an alternating minimization algorithm (Karakos et al., 2007a).

4 Experimental Methods and Results

We demonstrate the feasibility of cross-instance tuning with experiments on unsupervised document categorization from the 20 Newsgroups corpus (Lang, 1995); this corpus consists of roughly 20,000 news articles, evenly divided among 20 Usenet groups.

Random samples of 500 articles each were chosen by (Slonim et al., 2002) to create multiple test collections: 250 each from 2 arbitrarily chosen Usenet

groups for the *Binary* test collection, 100 articles each from 5 groups for the *Multi5* test collection, and 50 each from 10 groups for the *Multi10* test collection. Three independent test collections of each kind (*Binary*, *Multi5* and *Multi10*) were created, for a total of 9 collections. The sIB method was used to separately cluster each collection, given the correct number of clusters.

A comparison of sIB and IDTs on the *same* 9 test collections was reported in (Karakos et al., 2007b; Karakos et al., 2007a). Matlab code from (Slonim, 2003) was used for the sIB experiments, while the parameterized mutual information measures of Section 3 were used for the IDTs. A comparison was also made with the EM-based Gaussian mixtures clustering tool *mclust* (Fraley and Raftery, 1999), and with a simple K-means algorithm. Since the two latter techniques gave uniformly worse clusterings than those of sIB and IDTs, we omit them from the following discussion.

To show that our methods work beyond the 9 particular 500-document collections described above, in this paper we instead use five *different* randomly sampled test collections for each of the *Binary*, *Multi5* and *Multi10* cases, making for a total of 15 new test collections in this paper. For diversity, we ensure that none of the five test collections (in each case) contain any documents used in the three collections of (Slonim et al., 2002) (for the same case).

We pre-process the documents of each test collection using the procedure[2] mentioned in (Karakos et al., 2007b). The 15 test collections are then converted to feature matrices—term-document frequency matrices for sIB, and discounted tf/idf matrices (according to the Okapi formula (Gatford et al., 1995)) for IDTs—with each row of a matrix representing one document in that test collection.

[2]Excluding the subject line, the header of each abstract is removed. Stop-words such as *a*, *the*, *is*, etc. are removed, and stemming is performed (e.g., common suffixes such as -ing, -er, -ed, etc., are removed). Also, all numbers are collapsed to one symbol, and non-alphanumeric sequences are converted to whitespace. Moreover, as suggested in (Yang and Pedersen, 1997) as an effective method for reducing the dimensionality of the feature space (number of distinct words), all words which occur fewer than t times in the corpus are removed. For the sIB experiments, we use $t = 2$ (as was done in (Slonim et al., 2002)), while for the IDT experiments we use $t = 3$; these choices result in the best performance for each method, respectively, on another dataset.

4.1 Selecting α with "Strapping"

In order to pick the value of the parameter α for each of the sIB and IDT test experiments, we use "strapping" (Eisner and Karakos, 2005), which, as we mentioned earlier, is a technique for training a meta-classifier that chooses among possible clusterings. The training is based on unrelated instances of the same clustering task. The final choice of clustering is still unsupervised, since no labels (or ground truth, in general) for the instance of interest are used.

Here, our collection of possible clusterings for each test collection is generated by varying the α parameter. Strapping does not care, however, how the collection was generated. (In the original strapping paper, for example, Eisner and Karakos (2005) generated their collection by bootstrapping word-sense classifiers from 200 different seeds.)

Here is how we choose a particular unsupervised α-clustering to output for a given test collection:

- We cluster the test collection (e.g., the first Multi5 collection) with various values of α, namely $\alpha = 0.1, 0.2, \ldots, 1.0$.
- We compute a feature vector from each of the clusterings. Note that the features are computed from only the clusterings and the data points, since no labels are available.
- Based on the feature vectors, we predict the "goodness" of each clustering, and return the "best" one.

How do we predict the "goodness" of a clustering? By first learning to distinguish good clusterings from bad ones, by using unrelated instances of the task on which we know the true labels:

- We cluster some unrelated datasets with various values of α, just as we will do in the test condition.
- We evaluate each of the resulting clusterings using the true labels on its dataset.[3]
- We train a "meta-classifier" that predicts the true rank (or accuracy) of each clustering based on the feature vector of the clustering.

[3]To evaluate a clustering, one only really needs the true labels on a *sample* of the dataset, although in our experiments we did have true labels on the entire dataset.

Specifically, for each task (Binary, Multi5, and Multi10) and each clustering method (sIB and IDT), a meta-classifier is learned thus:

- We obtain 10 clusterings ($\alpha = 0.1, 0.2, \ldots, 1.0$) for each of 5 unrelated task instances (datasets whose construction is described below).
- For each of these 50 clusterings, we compute the following 14 features: (i) One minus the average cosine of the angle (in tf/idf space) between each example and the centroid of the cluster to which it belongs. (ii) The average Rényi divergence, computed for parameters $1.0, 0.5, 0.1$, between the empirical distribution of each example and the centroid of the cluster to which it belongs. (iii) We create 10 more features, one per α. For the α used in this clustering, the feature value is equal to $e^{-0.1\bar{r}}$, where $\bar{r}$ is the average rank of the clustering (i.e., the average of the 4 ranks resulting from sorting all 10 clusterings (per training example) according to one of the 4 features in (i) and (ii)). For all other α's, the feature is set to zero. Thus, only α's which yield relatively good rankings can have non-zero features in the model.
- We normalize each group of 10 feature vectors, translating and scaling each of the 14 dimensions to make it range from 0 to 1. (We will do the same at test time.)
- We train ranking SVMs (Joachims, 2002), with a Gaussian kernel, to learn how to rank these 50 clusterings given their respective normalized feature vectors. The values of c, γ (which control regularization and the Gaussian kernel) were optimized through leave-one-out cross validation in order to maximize the average accuracy of the top-ranked clustering, over the 5 training sets. Once a local maximum of the average accuracy was obtained, further tuning of c, γ to maximize the Spearman rank correlation between the predicted and true ranks was performed.

A model trained in this way knows something about the task, and may work well for many new, unseen instances of the task. However, we presume that it will work best on a given test instance if trained on similar instances. The ideal would be to match the test collection in every aspect: (i) the number of training labels should be equal to the number of desired clusters of the test collection; (ii) the training clusters should be topically similar to the desired test clusters.

In our scenario, we enjoy the luxury of plenty of labeled data that can be used to create similar instances. Thus, given a test collection $\mathcal{A}$ to be clustered into L clusters, we create similar training sets by identifying the L training newsgroups whose centroids in tf/idf space (using the Okapi formula mentioned earlier) have the smallest angle to the centroid of $\mathcal{A}$.[4] (Of course, we exclude newsgroups that appear in $\mathcal{A}$.) We then form a supervised 500-document training set $\mathcal{A}'$ by randomly choosing $500/L$ documents from each of these L newsgroups; we do this 5 times to obtain 5 supervised training sets.

Table 1 shows averaged classification errors resulting from strapping (*"str"* rows) for the Jensen-Rényi divergence and Csiszár's mutual information, used within IDTs and sIB, respectively. (We also tried the reverse, using Jensen-Rényi in sIB and Csiszár's in IDTs, but the results were uniformly worse in the former case and no better in the latter case.) The "MI" rows show the classification errors of the untuned algorithms ($\alpha = 1$), which, in almost all cases, are worse than the tuned ones.

4.2 Tuning α on Statistically Similar Examples

We now show that strapping outperforms a simpler and more obvious method for cross-instance tuning. To cluster a test collection $\mathcal{A}$, we could simply tune the clustering algorithm by choosing the α that works best on a related task instance.

We again take care to construct a training instance $\mathcal{A}'$ that is closely related to the test instance $\mathcal{A}$. In fact, we take even greater care this time. Given $\mathcal{A}$,

[4] For each of the Binary collections, the closest training newsgroups in our experiments were *talk.politics.guns, talk.religion.misc*; for each of the Multi5 collections the closest newsgroups were *sci.electronics, rec.autos, sci.med, talk.politics.misc, talk.religion.misc*, and for the Multi10 collections they were *talk.politics.misc, rec.motorcycles, talk.religion.misc, comp.graphics, comp.sys.ibm.pc.hardware, rec.sport.baseball, comp.os.ms-windows.misc, comp.windows.x, soc.religion.christian, talk.politics.mideast*. Note that each of the Binary test collections happens to be closest to the *same* two training newsgroups; a similar behavior was observed for the Multi5 and Multi10 newsgroups.

	Method \ Set	Binary	Multi5	Multi10
IDTs	MI	11.3%	9.9%	42.2%
	I_α *(str)*	**10.4%**	**9.2%**	**39.0%**
	I_α *(rls)*	**10.1%**	10.4%	42.7%
sIB	MI	12.0%	6.8%	38.5%
	I_α^C *(str)*	**11.2%**	6.9%	**35.8%**
	I_α^C *(rls)*	**11.1%**	7.4%	**37.4%**

Table 1: Average classification errors for IDTs and sIB, using strapping (*"str"* rows) and regularized least squares (*"rls"* rows) to pick α in Jensen-Rényi divergence and Csiszár's mutual information. Rows "MI" show the errors resulting from the *untuned* algorithms, which use the regular mutual information objective ($\alpha = 1$). Results which are better than the corresponding "MI" results are shown in **bold**.

we identify the same set of L closest newsgroups as described above. This time, however, we carefully select $|\mathcal{A}|/L$ documents from each newsgroup rather than randomly choosing $500/L$ of them. Specifically, for each test example (document) $X \in \mathcal{A}$, we add a similar training example X' into $\mathcal{A}'$, chosen as follows:

We associate each test example X to the most similar of the L training newsgroups, under a constraint that only $|\mathcal{A}|/L$ training examples may be associated to each newsgroup. To do this, we iterate through all pairs (X, G) where X is a test example and G is a training newsgroup, in increasing order by the angle between X and G. If X is not yet associated and G is not yet "full," then we associate X with G, and choose X' to be the document in G with the smallest angle to X.

We cluster $\mathcal{A}'$ 10 times, for $\alpha = 0.1, \ldots, 1.0$, and we collect supervised error results $\mathcal{E}(\alpha),\ \alpha \in \{0.1, \ldots, 1.0\}$. Now, instead of using the single best $\alpha^* = \operatorname{argmin}_\alpha \mathcal{E}(\alpha)$ to cluster $\mathcal{A}$ (which may result in overfitting) we use regularized least-squares (RLS) (Hastie et al., 2001), where we try to approximate the *probability that an α is the best*. The estimated probabilities are given by

$$\hat{\boldsymbol{p}} = \mathbf{K}(\lambda \mathbf{I} + \mathbf{K})^{-1}\boldsymbol{p},$$

where $\mathbf{I}$ is the unit matrix, $\boldsymbol{p}$ is the training probability of the best α (i.e., it is 1 at the position of α^* and zero elsewhere), and $\mathbf{K}$ is the kernel matrix, where $K(i,j) = \exp(-(\mathcal{E}(\alpha_i) - \mathcal{E}(\alpha_j))^2/\sigma^2)$ is the value of the kernel which expresses the "similarity" between two clusterings of the same training dataset, in terms of their errors. The parameters σ, γ are set to $0.5, 0.1$, respectively, after performing a (local) maximization of the Spearman correlation between training accuracies and predicted probabilities $\hat{\boldsymbol{p}}$, for all 15 training instances. After performing a linear normalization of $\hat{\boldsymbol{p}}$ to make it a probability vector, the average predicted value of α, i.e., $\hat{\alpha} = \sum_{i=1}^{10} \hat{p}_i\, \alpha_i$, (rounded-off to one of $\{0.1, \ldots, 1.0\}$) is used to cluster $\mathcal{A}$.

Table 1 shows the average classification error results using RLS (*"rls"* rows). We can see that, on average over the 15 test instances, the error rate of the tuned IDTs and sIB algorithms is lower than that of the untuned algorithms, so cross-instance tuning was effective. On the other hand, the errors are generally higher than that of the strapping method, which examines the results of using different α values on $\mathcal{A}$.

5 Concluding Remarks

We have considered the problem of cross-instance tuning of two unsupervised document clustering algorithms, through the introduction of a degree of freedom into their mutual information objective. This degree of freedom is tuned using *labeled* document collections (which are unrelated to the test collections); we explored two approaches for performing the tuning: (i) through a judicious sampling of training data, to match the marginal statistics of the test data, and (ii) via "strapping", which trains a meta-classifier to distinguish between good and bad clusterings. Our unsupervised categorization experiments from the "20 Newsgroups" corpus indicate that, although both approaches improve the baseline algorithms, "strapping" is clearly a better choice for knowledge transfer between unrelated task instances.

References

R. K. Ando and T. Zhang. 2005. A framework for learning predictive structures from multiple tasks and unlabeled data. *Journal of Machine Learning Research*, 6:1817–1853, Nov.

S. Ben-David and R. Schuller. 2003. Exploiting task relatedness for multiple task learning. In *Proc. of the Sixteenth Annual Conference on Learning Theory (COLT-03)*.

A. Blum and T. Mitchell. 1998. Combining labeled and unlabeled data with co-training. In *Proceedings of the Workshop on Computational Learning Theory (COLT-98)*, pages 92–100.

R. Caruana. 1997. Multitask learning. *Machine Learning*, 28(1):41–75.

T. Cover and J. Thomas. 1991. *Elements of Information Theory*. John Wiley and Sons.

I. Csiszár. 1995. Generalized cutoff rates and Rényi's information measures. *IEEE Trans. on Information Theory*, 41(1):26–34, January.

I. Dhillon, S. Mallela, and R. Kumar. 2003. A divisive information-theoretic feature clustering algorithm for text classification. *Journal of Machine Learning Research (JMLR), Special Issue on Variable and Feature Selection*, pages 1265–1287, March.

J. Eisner and D. Karakos. 2005. Bootstrapping without the boot. In *Proc. 2005 Conference on Human Language Technology / Empirical Methods in Natural Language Processing (HLT/EMNLP 2005)*, October.

T. Evgeniou and M. Pontil. 2004. Regularized multi-task learning. In *Proc. Knowledge Discovery and Data Mining*.

C. Fraley and A. E. Raftery. 1999. Mclust: Software for model-based cluster analysis. *Journal on Classification*, 16:297–306.

M. Gatford, M. M. Hancock-Beaulieu, S. Jones, S. Walker, and S. E. Robertson. 1995. Okapi at TREC-3. In *The Third Text Retrieval Conference (TREC-3)*, pages 109–126.

A. Ben Hamza and H. Krim. 2003. Jensen-Rényi divergence measure: Theoretical and computational perspectives. In *Proc. IEEE Int. Symp. on Information Theory*, Yokohama, Japan, June.

T. Hastie, R. Tibshirani, and J. Friedman. 2001. *The Elements of Statistical Learning*. Springer-Verlag.

A. O. Hero, B. Ma, O. Michel, and J. Gorman. 2001. Alpha-divergence for classification, indexing and retrieval. Technical Report CSPL-328, University of Michigan Ann Arbor, Communications and Signal Processing Laboratory, May.

F. Jelinek. 1997. *Statistical Methods for Speech Recognition*. MIT Press.

T. Joachims. 2002. Optimizing search engines using clickthrough data. In *ACM Conf. on Knowledge Discovery and Data Mining (KDD)*.

D. Karakos, S. Khudanpur, J. Eisner, and C. E. Priebe. 2005. Unsupervised classification via decision trees: An information-theoretic perspective. In *Proc. 2005 International Conference on Acoustics, Speech and Signal Processing (ICASSP 2005)*, March.

D. Karakos, S. Khudanpur, J. Eisner, and C. E. Priebe. 2007a. Information-theoretic aspects of iterative denoising. Submitted to the Journal of Machine Learning Research, February.

D. Karakos, S. Khudanpur, J. Eisner, and C. E. Priebe. 2007b. Iterative denoising using Jensen-Rényi divergences with an application to unsupervised document categorization. In *Proc. 2007 International Conference on Acoustics, Speech and Signal Processing (ICASSP 2007)*, April.

K. Lang. 1995. Learning to filter netnews. In *Proc. 13th Int. Conf. on Machine Learning*, pages 331–339.

David D. Lewis and Philip J. Hayes. 1994. Guest editorial. *ACM Transactions on Information Systems*, 12(3):231, July.

C. E. Priebe, D. J. Marchette, and D. M. Healy. 2004a. Integrated sensing and processing decision trees. *IEEE Trans. on Pat. Anal. and Mach. Intel.*, 26(6):699–708, June.

C. E. Priebe, D. J. Marchette, Y. Park, E. Wegman, J. Solka, D. Socolinsky, D. Karakos, K. Church, R. Guglielmi, R. Coifman, D. Lin, D. Healy, M. Jacobs, and A. Tsao. 2004b. Iterative denoising for cross-corpus discovery. In *Proc. 2004 International Symposium on Computational Statistics (COMPSTAT 2004)*, August.

N. Slonim and N. Tishby. 2000. Document clustering using word clusters via the information bottleneck method. In *Research and Development in Information Retrieval*, pages 208–215.

N. Slonim, N. Friedman, and N. Tishby. 2002. Unsupervised document classification using sequential information maximization. In *Proc. SIGIR'02, 25th ACM Int. Conf. on Research and Development of Inform. Retrieval*.

N. Slonim. 2003. IBA_1.0: Matlab code for information bottleneck clustering algorithms. Available from http://www.princeton.edu/∼nslonim/IB_Release1.0/IB_Release1_0.tar.

N. Tishby, F. Pereira, and W. Bialek. 1999. The information bottleneck method. In *37th Allerton Conference on Communication and Computation*.

K. Torkkola. 2002. On feature extraction by mutual information maximization. In *Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Proc. (ICASSP-2002)*, May.

Y. Yang and J. Pedersen. 1997. A comparative study on feature selection in text categorization. In *Intl. Conf. on Machine Learning (ICML-97)*, pages 412–420.

D. Yarowsky. 1995. Unsupervised word sense disambiguation rivaling supervised methods. In *Proc. 33rd Annual Meeting of the Association for Computational Linguistics*, pages 189–196, Cambridge, MA.