Discriminative Clustering for Market Segmentation

Peter Haider[*]
Dep. of Computer Science
Univ. of Potsdam, Germany
haider@cs.uni-potsdam.de

Luca Chiarandini[†]
Web Research Group
Universitat Pompeu Fabra
Barcelona, Spain
chiarluc@yahoo-inc.com

Ulf Brefeld
University of Bonn
Bonn, Germany
brefeld@uni-bonn.de

ABSTRACT

We study discriminative clustering for market segmentation tasks. The underlying problem setting resembles discriminative clustering, however, existing approaches focus on the prediction of univariate cluster labels. By contrast, market segments encode complex (future) behavior of the individuals which cannot be represented by a single variable. In this paper, we generalize discriminative clustering to structured and complex output variables that can be represented as graphical models. We devise two novel methods to jointly learn the classifier and the clustering using alternating optimization and collapsed inference, respectively. The two approaches jointly learn a discriminative segmentation of the input space and a generative output prediction model for each segment. We evaluate our methods on segmenting user navigation sequences from Yahoo! News. The proposed collapsed algorithm is observed to outperform baseline approaches such as mixture of experts. We showcase exemplary projections of the resulting segments to display the interpretability of the solutions.

Categories and Subject Descriptors

I.5.3 [**Clustering**]: Algorithms

General Terms

Algorithms, Experimentation

Keywords

Discriminative Clustering, Market Segmentation

1. INTRODUCTION

Market segmentation reveals divisions in a given market, where a market refers to a population of interest such as people, customers, or organizations. A market segment is a subset of a market that is characterized by similar demands and/or needs based on qualities of a given product such as price or function.

Every segment is required to meet the following criteria. (i) It is homogeneous within the segment, so that individuals within the same segment exhibit common needs and can be targeted jointly with the same marketing strategy. (ii) It is easily distinguishable from other segments to guarantee that different segments have different demands and (iii) serves as a blueprint for distinct targeting strategies. The requirements are often summarized as *homogeneity*, *identifiability*, and *interpretability* (19).

Besides frequently deployed self-organizing maps (SOMs) (13; 7), market segmentation matches the problem setting of model-based clustering approaches. Clustering techniques either minimize the within-cluster similarity (1; 14), maximize the between-cluster similarity (7), or optimize a combination of both (18), and thus aim to produce homogeneous and identifiable solutions. Once segments have been computed, new customers need to be assigned to one of the subgroups to advertise the product accordingly.

Unfortunately, mapping new instances to an existing clustering is often difficult in practice. Intuitively, the new customer should be grouped to the closest segment with respect to some similarity measure. Closeness can for instance be computed as the distance in feature space to the median or the nearest member of a segment (11; 4). However, often at the time of classifying a new instance, not all features are known. Even using the similarity measure that is used by the clustering method itself on all features is frequently observed to perform surprisingly inaccurate, see e.g., (20) and Section 4. Other difficulties are for instance distribution-based clusterings, such as Expectation Maximization (6), which assign probabilities for cluster-memberships. By doing so, customers are probabilistically related to *every* segment. Converting this soft assignment into a hard assignment by taking a *maximum a posteriori* or *winner-takes-all* decision is often suboptimal if the memberships deviate from a point distribution, in which case more than one segments are likely candidates (14).

Optimally, the segmentation is therefore learned together with a classifier that discriminatively maps new instances to clusters; a problem setting which is also known as discriminative clustering (5; 20; 24; 8). The idea is to have the clustering provide the labels for the classifier which is trained in a supervised manner. The joint optimization alters the clustering so that the segments can be easily dis-

[*]This work was performed during an internship at Yahoo! Research, Barcelona, Spain.

[†]Also Yahoo! Research, Barcelona, Spain.

criminated from each other by the classifier. Combining the two criteria thus guarantees concise clusterings and accurate classifiers. Existing approaches focus on clustering a population and predicting a cluster label for a new instance. By contrast, market segmentation is more complex. In market segmentation tasks, we need to differentiate between the data that characterizes individuals and the data that characterizes their future behavior. The clustering clearly needs to take all available information into account to generate meaningful segments. However, the classifier does not have access to future events and needs to take a decision on the available information such as gender, income, etc. This observation renders existing approaches to discriminative clustering too restrictive for market segmentation tasks.

In this paper we generalize discriminative clustering for market segmentation tasks using the structured prediction framework. We differentiate between *attributes* of a customer and her *interests/behavior*. *Attributes* are *a priori* available features of individuals of the population such as gender or income. Her *behavior* is a collection of interacting variables describing a segment. As segments need to be interpretable, we model the output data as a complex and structured variable which can be represented as a graphical model. The distinction allows for learning a classifier only on the attributes, computing the clustering on both attributes and behavior, and finally summarizing the segments only in terms of the behavior.

We devise two solutions which are based on the regularized empirical risk minimization framework. The first is a straightforward adaptation of mixtures of experts. Classifier and clustering are optimized using an alternating strategy where we fix one component while optimizing the other. The second solution uses approximations and integrates out parameters of the classifier using collapsed inference for efficiency. Both approaches use generative models for the output structure and, in contrast to conventional discriminative clustering approaches, do not involve trade-off parameters for classification accuracy and cluster consistency (class balance) because the optimization problems are not prone to trivial and degenerate solutions.

Use cases of our methods contain traditional market segmentation tasks. Consider for instance a company that aims at promoting a new product or a hotel chain that intends to lure visitors with special offers. Our methods not only compute a meaningful segmentation of the customers but also allow for devising appropriate targeting strategies from the graphical models. Moreover, our method serves as discriminative clustering for structured variables, where the task is not to output a single class/cluster label but the average structure for every segment. The differentiation between attributes and behavior increases the range of applications that can be addressed. A special – but still novel – case is obtained when attributes and behavior partially overlap.

Empirically, we study our methods on another interesting use case: Segmenting user navigation sessions on the Web for displaying segment-specific website layouts. We experiment on a large click log from Yahoo! News. The attribute data is assembled from meta-information about the session such as the timestamp, the referrer domain, and the first page request. The behavior consists of subsequent navigation actions given by click sequences. The generative representation of the behavior data is interpretable and can be easily transformed into segment-specific layouts.

The remainder of the paper is structured as follows. Section 2 discusses the relationship of our problem setting with previously studied settings and methods. In Section 3 we derive two algorithms to optimize the empirical counterpart of the expected segmented log-likelihood. Section 4 reports on empirical results using a large click log from a commercial news provider and Section 5 concludes.

2. RELATED WORK

Market segmentation tasks are often solved using neural networks such as self-organizing maps (13; 7). Kiang et al. (13) for instance extend self-organizing maps to group customers according to their attitude towards different communications modes. D'Urso and de Giovanni (7) use the natural clustering property of self-organizing maps together with dissimilarity measures which capture temporal structure of the data. In general, clustering data with self-organizing maps and variants thereof inherently implements the homogeneity assumption of market segmentation. However, classifying new instances into the clustering is often difficult and it is not possible to output generative models to summarize the resulting clusters. Additionally, the optimization criterion of self-organizing maps is highly sensitive to the actual initialization and usually converges to different local optima.

Related to market segmentation is the task of estimating a mixture model for observations (6). Introducing selector variables encoding probabilistic cluster-memberships, maximizing the log-likelihood by marginalizing over the selector is usually straightforward. The selector can be modeled in a data-dependent or data-independent fashion but the probabilistic nature of the cluster-memberships render a direct application for market segmentation tasks impossible.

Discriminative clustering simultaneously computes a segmentation of the data at hand *and* a classifier that discriminates the resulting clusters well. Existing approaches include projections into lower-dimensional subspaces (5), joint optimization of max-margin classifiers and clusterings (20; 24), the optimization of scatter metrics (21), and the maximization of an information theoretic criterion to balance class separation and classifier complexity (8). Sinkkonen et al. (17) aim to find clusters that are homogeneous in auxiliary data given by additional discrete variables. The above mentioned approaches do not predict any output variable but focus on the discrete cluster variable. Moreover, in our generalized problem setting, instances are represented as input-output pairs. The classifier discriminates the clusters given only the input, whereas the cluster parameters need to accurately estimate the outputs of the contained instances. Previous work on discriminative clustering does not split instances into two parts. They represent instances as a single input which consequently allows the classifiers to access the whole example at decision time. The same assumption is commonly being made in market segmentation studies that involve model-based clustering approaches, (9; 16; 22) but prohibits a natural solution for market segmentation tasks.

This problem setting can be seen as an alteration of the setting which the Mixture of Experts approach (10; 12) aims to solve, where the behavior y is predicted given the attributes x as a mixture model where the mixture component weights depend again on x. In our case, mixture component weights have to be always point distributions as demanded by the application. Framing the distribution of y given the mixture component as a pure generative model allows us to derive a more efficient algorithm than that of the Mixture of Experts approach.

Zhao et al. (23) proposed a maximum-margin clustering for multivariate loss functions. Minimizing the complex losses allows for capturing structural differences that cannot be expressed in terms of standard misclassification rates. In principle, by defining a loss function that captures the differences of two clusterings, one could possibly solve market segmentation tasks as their approach implicitly favors clusterings that are easily discriminable. However, the loss function cannot be expressed in terms of the contingency table of the two clusterings, and the decoding problem in the inner loop of the algorithm, that is finding the most violated constraint, becomes intractable in practice.

Also related to our problem setting are multi-view clustering approaches, where the data is split into two disjoint feature sets, which are sometimes also called views. Bickel and Scheffer (2) present an intertwined Expectation Maximization algorithm to compute the maximum likelihood solution for one view using the expectations provided by its peer view. The two data views are modeled generatively and the algorithm maximizes the joint marginal likelihood. By contrast, we aim to find a discriminative classifier on the input view and instead of maximizing the marginal likelihood of the output view we seek to maximize the likelihood conditioned on a hard cluster assignment.

3. DISCRIMINATIVE SEGMENTATION

We now present our main contribution, the generalization of discriminative clustering for structured output variables to solve market segmentation problems. We introduce the problem setting in the next section and present a straightforward solution in terms of mixtures of experts in Section 3.2. An efficient approximation is devised in Section 3.3 and Section 3.3.3 discusses scalability issues.

3.1 Preliminaries

We are given a sample $\mathcal{M}$ from a market where individuals are represented by tuples $(x, y) \in \mathcal{X} \times \mathcal{Y}$ encoding attributes x and behavior y. Attributes x may encompass individual features like gender, income, etc while the expressed historic behavior is captured by $y \in \mathcal{Y}$ and represented as a graphical model. The behaviors y are governed by a family of distributions denoted by $P(y|\theta)$ with natural parameter θ.

In our running example on segmenting user navigation on the Web, attributes x encode meta-information about the session such as the timestamp, the referrer domain, and the first page request and is represented as a feature vector. The behavior y encodes sequences of the subsequent Web navigation and can for instance be represented as a Markov-chain where nodes correspond to pageviews and connecting edges visualize clicks.

We aim to find an appropriate segmentation of the market $\mathcal{M}$. Formally, the goal is to find a classifier $h : \mathcal{X} \to \{1, \ldots, k\}$ that maps attributes x to one of k clusters, parameterized by $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_k)$, where the θ_j are chosen to maximize the likelihood of the observed behaviors y of the respective segment. The number of clusters k is assumed to be given by the application at hand, because it constitutes a trade-off between predictive power and effort spent for developing multiple market strategies. As h and $\boldsymbol{\theta}$ are not independent they need to be optimized jointly. Hence, over all classifiers h and parameter collections $\boldsymbol{\theta}$, we aim at maximizing the expected risk functional R that is defined in terms of the segmented log-likelihood

$$R(h, \boldsymbol{\theta}) = \int \log P(y|\theta_{h(x)}) dP(x, y). \tag{1}$$

Since the true joint distribution $P(x, y)$ is unknown, we replace Equation (1) by its empirical counterpart on the finite market sample of size n given by $\mathcal{M} = \{(x_i, y_i)\}_{i=1}^n$

$$\hat{R}(\boldsymbol{\theta}, h) = \sum_{i=1}^{n} \log P(y_i|\theta_{h(x_i)}). \tag{2}$$

Directly maximizing Equation (2) in terms of the component parameters $\boldsymbol{\theta}$ and the classifier h is infeasible, since the objective function is not only highly non-convex and non-continuous but an NP-hard problem because of combinatorial assignments. However, if the classifier h was fixed, the θ_j could be optimized directly, as h provides the segmentation and for each segment j optimal parameters $\hat{\theta}_j$ are trivially computed by

$$\hat{\theta}_j = \operatorname*{argmax}_{\theta} \sum_{i:h(x_i)=j} \log P(y_i|\theta). \tag{3}$$

For many common distribution families, the maximum likelihood estimates $P(y|\theta)$ can be computed easily by counting or averaging over the observations in the segment, respectively. Vice versa, keeping the segment parameters $\theta_1, \ldots, \theta_k$ fixed, learning the classifier h results in a standard multi-class classification scenario. Using linear models, h can be written as

$$h(x) = \operatorname*{argmax}_{j \in \{1,\ldots,k\}} w_j^\top x, \tag{4}$$

where each segment has its own weight vector w_j. In the remainder, we will use h and $\boldsymbol{w} = (w_1, \ldots, w_k)^\top$ interchangeably. The next section exploits this observation and presents a joint alternating optimization scheme.

3.2 An Alternating Optimization Scheme

A straightforward approach to solve market segmentation problems is to alternate the optimization of the classifier and the clustering while fixing the other, respectively. As shown in Equation (3), keeping the classifier fixed allows to apply standard maximum likelihood techniques to compute the natural parameters of the segments. We thus focus on deriving the classifier h for a fixed clustering. We make use of the maximum-margin framework and deploy a rescaled variant of the hinge loss to deal with uncertain cluster-memberships (or class labels).

The idea is as follows. Intuitively, an individual (x, y) should be assigned to the segment that realizes the highest log-likelihood with respect to y. However, two or more segments might be competing for the instance and realize

Algorithm 1 Alternating Optimization
```
1: Input: $(x_1, y_1), \ldots, (x_n, y_n), \lambda > 0, k > 1$
2: Initialize $\boldsymbol{\theta}$ randomly
3: repeat
4:    E-step: $\boldsymbol{w} \leftarrow \operatorname{argmin}_{\boldsymbol{w}', \boldsymbol{\xi} \geq \boldsymbol{0}} \; \frac{\lambda}{2}\|\boldsymbol{w}'\|^2 + \frac{1}{n}\sum_{i=1}^{n} \xi_i$
5:                        s.t.   $(w'_{j_i^*} - w'_j)^\top x \geq 1 - \frac{\xi_i}{s(y)}$
6:    M-step: $\boldsymbol{\theta} \leftarrow \operatorname{argmax}_{\boldsymbol{\theta}'} \sum_j \sum_{i:h(x_i)=j} \log P(y_i|\theta')$
7: until convergence
```

similar log-likelihoods, in which case a winner-takes-all decision is prohibitive. We thus treat the difference of the log-likelihoods between the most likely segment j^* and cluster $j' \neq j^*$ as a *misclassification score*, given by

$$s(y) = \log P(y|\theta_{j^*}) - \log P(y|\theta_{j'}). \quad (5)$$

These scores can be incorporated in a support vector machine by re-scaling the hinge loss and act like example-dependent costs (3). The re-scaled hinge loss becomes a convex upper bound of the difference of the log-likelihoods,

$$\ell(x) = s(y) \max\left(0, 1 - (w_{j^*} - w_{j'})^\top x\right). \quad (6)$$

Stacking up $\boldsymbol{w} = (w_1, \ldots, w_k)^\top$ and $\boldsymbol{\xi} = (\xi_1, \ldots, \xi_n)^\top$, we arrive at the following maximum-margin optimization problem

$$\min_{\boldsymbol{w}, \boldsymbol{\xi} \geq \boldsymbol{0}} \quad \frac{\lambda}{2}\|\boldsymbol{w}\|^2 + \frac{1}{n}\sum_{i=1}^{n} \xi_i$$

$$\text{s.t.} \quad (w_{j_i^*} - w_j)^\top x \geq 1 - \frac{\xi_i}{s(y)},$$

for $1 \leq i \leq n$, $j \in \{1, \ldots, k\}$, $j \neq j^*$, $j_i^* = \arg\max_j P(y_i|\theta_j)$, and regularizaion parameter $\lambda > 0$.

Hence, Equation (2) can be optimized by randomly initializing $\boldsymbol{\theta}$ and subsequently alternating between optimizing $\boldsymbol{w}$ and θ_j while fixing the respective peer. Algorithm 1 instantiates the pseudo-code as a member of the Expectation Maximization framework. The alternating optimization scheme can also be interpreted as an adaptation of the EM algorithm for a mixture of experts (12). The classifier h, given as the weight vectors w_j, corresponds to the gating networks. In contrast to the conventional mixture of experts model, it is trained using an SVM to output deterministic decisions, instead of soft decisions. The generative distribution over behaviors differs from the expert networks only in the fact that in our problem setting the prediction of the behavior y is not allowed to depend directly on the attributes x, only via the cluster assignment.

A major drawback of the alternating approach is however the discrete assignment of individuals to only a single segment, even during the intermediate optimization steps. As a consequence, the algorithm is prone to degenerate solutions and poor local optima. Additionally, the optimization is expensive in terms of computational time as it requires the computation of a multi-class SVM until convergence in every step.

3.3 Collapsed Optimization

We now present an efficient approximation of the discriminative segmentation problem by using a continuous relaxation to the original problem formulation. We first show that the parameters of the classifier can be computed in closed-form so that the joint optimization problem depends only on the segment parameters. Second, we devise an EM-like algorithm (6) to optimize the remaining objective.

3.3.1 Eliminating the Classifier Parameters

As the discrete cluster assignments cause many difficulties, we now replacing them by soft assignments using an adjustable soft-max function. The parameter ρ controls the degree of reproducing the maximum, that is for $\rho \to \infty$ we precisely obtain the maximum operator. Incorporating the soft-max in Equation (2) yields the optimization problem

$$\max_{\boldsymbol{\theta}, \boldsymbol{w}} \sum_{i=1}^{n} \log \sum_{j=1}^{k} P(y_i|\theta_j) \frac{\exp(\rho\, w_j^\top x_i)}{\sum_{j'} \exp(\rho\, w_{j'}^\top x_i)}, \quad (7)$$

which still contains the mutually dependent variables $\boldsymbol{\theta}$ and $\boldsymbol{w}$. To obtain an efficiently solvable optimization problem, we express the objective as a continuous function of $\boldsymbol{w}$ so that $\boldsymbol{w}$ can be eliminated using collapsed inference. Instead of the hinge loss in Equation (6), we employ another tight convex upper bound in terms of the squared loss,

$$\ell(x) = (\log P(y|\theta_j) - w_j^\top x)^2.$$

Implicitly, introducing the squared loss converts the classifier into a regressor that aims at predicting the log-likelihood for an individual (x, y) and the j-th segment as accurate as possible. Assuming the log-likelihoods were predicted perfectly, the parameters $\boldsymbol{w}$ would not only be optimal for the regression but also for Equation (2) as the classifier h in Equation (4) would still return the most likely segment. Changing the loss function also has the advantage that now the optimal solution for $\boldsymbol{w}$ can be computed analytically. The corresponding regularized optimization problem is also known as regularized least squares regression (RLSR) or ridge regression and is given by

$$\min_{\boldsymbol{w}} \frac{\lambda}{2}\|\boldsymbol{w}\|^2 + \sum_{i=1}^{n}\sum_{j=1}^{k}\left(\log P(y_i|\theta_j) - w_j^\top x_i\right)^2, \quad (8)$$

for $\lambda > 0$. Setting the derivative with respect to $\boldsymbol{w}$ to zero and solving for w_j, we obtain the optimal solution that can be computed in closed-form

$$w_j = X^\top (XX^\top + \lambda I)^{-1} \pi(\theta_j), \quad (9)$$

where $X \in \mathbb{R}^{n \times d}$ contains the stacked attribute vectors and $\pi(\theta_j) = (\log P(y_1|\theta_j), \ldots, \log P(y_n|\theta_j))^\top$ is the vector of log-likelihoods for the j-th segment. The computation of the inner product

$$w_j^\top x = \pi(\theta_j)^\top (XX^\top + \lambda I)^{-1} X x$$

can effectively be sped-up by precomputing the linear transformation of the attributes. Introducing auxiliary variables $\bar{x}$ given by

$$\bar{x} = (XX^\top + \lambda I)^{-1} X x,$$

allows to efficiently rewrite the inner product as $w_j^\top x = \pi(\theta_j)^\top \bar{x}$. Further note that $\bar{x}$ depends only on inner products of individual attributes. Hence, the kernel trick can be applied to incorporate non-linearity. Introducing a Mercer kernel κ, the auxiliary matrix $\bar{X}$ can be written in terms of the corresponding kernel matrix K with elements $K_{ij} = \{\kappa(x_i, x_j)\}_{ij}$ as $(K + \lambda I)^{-1} K$. The classifier can then be expressed in terms of a set of dual multipliers α and the kernel function as $h(x) = \arg\max_j \sum_i \alpha_{ji} \kappa(x_i, x_j)$. The dual multipliers can be obtained explicitly as a function of the component parameters as

$$\alpha_j(\theta_j) = \pi(\theta_j)^\top (K + \lambda I)^{-1}. \quad (10)$$

Substituting the obtained observations in Equation (7) results in a simplified optimization problem that does no longer depend on $\boldsymbol{w}$ and that has only the θ_j as free parameters,

$$\max_{\boldsymbol{\theta}} \sum_{i=1}^{n} \log \sum_{j=1}^{k} P(y_i|\theta_j) \frac{\exp(\rho\,\pi(\theta_j)^\top \bar{x}_i)}{\sum_{j'} \exp(\rho\,\pi(\theta_j)^\top \bar{x}_i)}. \qquad (11)$$

3.3.2 Optimizing the segment parameters

The optimization problem in Equation (11) can be solved with an EM-like algorithm (6) using auxiliary variables $z_{i,j} \geq 0$, with $\sum_j z_{i,j} = 1$, encoding the belief that the i-th example belongs to the j-th cluster. EM-like algorithms consist of two phases which assign instances to generating segments (E-step) and maximize the segment parameters with respect to the associated instances (M-step), respectively. In the E-step, it therefore suffices to identify the auxiliary variables with the true posterior probabilities given the current $\boldsymbol{\theta}$,

$$z_{i,j} \propto P(y_i|\theta_j) \frac{\exp(\rho\,\pi(\theta_j)^\top \bar{x}_i)}{\sum_{j'} \exp(\rho\,\pi(\theta_j)^\top \bar{x}_i)}.$$

Following the general EM framework, we express the empirical risk functional in Equation (11) in terms of the expectations $z_{i,j}$. This allows us to effectively pull the logarithm into the sum over segments for the M-step; we arrive at the optimization problem

$$\max_{\boldsymbol{\theta}} \sum_i \sum_j z_{i,j} \log \left[P(y_i|\theta_j) \frac{\exp(\rho\,\pi(\theta_j)^\top \bar{x}_i)}{\sum_{j'} \exp(\rho\,\pi(\theta_j)^\top \bar{x}_i)} \right],$$

which can be rewritten as

$$\max_{\boldsymbol{\theta}} \sum_{i,j} z_{i,j} \left[\rho\pi(\theta_j)^\top \bar{x}_i - \log \sum_{j'} \exp(\rho\pi(\theta_{j'})^\top \bar{x}_i) + \log P(y_i|\theta_j) \right].$$

The above Equation is to be maximized with respect to $\boldsymbol{\theta}$. Compared to conventional M-steps for mixture model estimation problems, $\boldsymbol{\theta}$ appears not only in $P(y_i|\theta_j)$, but also in what are usually the segment weights for each example. This renders the objective function non-concave and, consequentially, there is no exact analytical solution.

As a remedy, we derive an approximation that is linear and non-decreasing in the $\pi(\theta_j)$, rendering the objective concave in θ_j and thus analytically solvable for common choices of $P(y|\theta_j)$. We begin with approximating the normalization term $Z(\boldsymbol{\theta}) = \log \sum_{j'} \exp(\rho\pi(\theta_{j'})^\top \bar{x}_i)$ by its first-order Taylor expansion around the current $\boldsymbol{\theta}^{old}$ which is given by

$$Z(\boldsymbol{\theta}) \approx \sum_{j'} t_{ij'} \rho\pi(\theta_{j'})^\top \bar{x}_i + C, \qquad (12)$$

where the $t_{ij'} = \frac{\exp(\rho\pi(\theta_{j'}^{old})^\top \bar{x}_i)}{\sum_j \exp(\rho\pi(\theta_j^{old})^\top \bar{x}_i)}$ are the Taylor coefficients and C is a constant. Substituting Equation (12) into the objective function of the M-step and collecting the coefficients gives us

$$\underset{\boldsymbol{\theta}}{\operatorname{argmax}} \sum_i \sum_j \log P(y_i|\theta_j) \left[z_{i,j} + \rho \sum_{i'} \bar{x}_{i'i} \Big(z_{i',j}) - \sum_{j'} z_{i',j'} t_{i'j'} \Big) \right], \qquad (13)$$

Algorithm 2 Collapsed Optimization Algorithm

```
1: Input: $(x_1, y_1), \ldots, (x_n, y_n), \lambda$
2: $\rho \leftarrow 1$, $t \leftarrow 1$, initialize $\boldsymbol{\theta}^{(0)}$ randomly
3: repeat
4:     E-step: $Q(z_i = j) \leftarrow P(z_i = j|x_i, y_i, \rho, \lambda, \boldsymbol{\theta}^{(t-1)})$
5:     M-step: $\boldsymbol{\theta}^{(t)} \leftarrow \arg\max_{\boldsymbol{\theta}} \sum_i \sum_j Q(z_i = j) \times$
6:                                 $\log P(y_i, z_i = j|x_i, \rho, \lambda, \boldsymbol{\theta})$
7:     $\rho \leftarrow \rho \times 1.1$, $t \leftarrow t + 1$
8: until convergence
9: $\hat{\boldsymbol{\theta}} = \operatorname{argmax}_{\boldsymbol{\theta}} \hat{R}(\boldsymbol{\theta}, \alpha(\boldsymbol{\theta}))$
10: $\alpha \leftarrow \alpha(\hat{\boldsymbol{\theta}})$
```

which is a linear function of $\log P(y_i|\theta_j)$. For increasing ρ, the Taylor coefficients for each instance approach a point distribution exponentially fast, i.e. $t_{i\cdot} \to (0, \ldots, 0, 1, 0, \ldots, 0)^\top$. The same holds for the auxiliary variables $z_{i,j}$, which approach t_{ij}. Thus, the second summands of the coefficients in Equation (13) approach zero, and hence for large ρ all the $\log P(y_i|\theta_j)$ have either positive or very small negative coefficients. We clip coefficients below zero to guarantee that the objective function is non-decreasing in the $\log P(y_i|\theta_j)$ and obtain an approximation that approaches the exact solution with increasing ρ.

The softmax factor ρ therefore constitutes a trade-off between accurately approximating the original optimization problem of maximizing $\hat{R}(\boldsymbol{\theta}, h)$ and smoothing the objective function to facilitate finding good local optima.

We deal with this tradeoff by starting the EM-algorithm with $\rho = 1$, and multiplying it by a constant factor each iteration. Preliminary experiments have shown the factor 1.1 to work well. Due to the approximation of the hard decisions with a soft-max, the algorithm is not guaranteed to monotonically increase the true objective value. We thus select the intermediate result with the highest objective value as the final solution for the cluster parameters.

Algorithm 2 shows the collapsed optimization algorithm in pseudo code. In line 2, the cluster parameters $\boldsymbol{\theta}$ are initialized randomly. Line 4 performs the expectation step of the EM-algorithm, computing the current posterior estimates, given the soft-max factor ρ, the cluster parameters, and implicitly also the classifier h. The maximization step in line 5 boils down to an optimization problem of the form $\sum_i \sum_j c_{ij} \log P(y_i|\theta_j)$, which for non-negative coefficients c_{ij} can be solved analytically for many choices of distribution families. For example if $P(y|\theta)$ is a multinomial distribution over $y \in \{1, \ldots, m\}$ with $P(y = q|\theta_j) = \theta_{j,q}$, the maximum is attained for $\theta_{j,q} = \sum_i c_{ij} [\![y_i = q]\!] / \sum_i c_{ij}$ for all j, q, where $[\![\cdot]\!]$ is the indicator function. Finally, lines 9 and 10 select the best intermediate solution in terms of the true objective, and re-obtain the explicit classifier using Equation (10).

3.3.3 Scalability

The computational bottlenecks of the collapsed optimization algorithm are the computation of the $\bar{x}_i$, involving matrix inversions and multiplications, and the computation of the coefficients in Equation (13), where we have to sum over all pairs of examples, leading to an overall complexity of the algorithm of $O(n^{2.376} + n^2 kT)$.

For applications with a large number of examples the super-quadratic dependence on the sample size n makes the algorithm effectively intractable. We can alleviate this by randomly partitioning the examples in the least-squares estimation in Equation (8) into s disjoint subsets $S^{(1)}, \ldots, S^{(s)}$ of size m. For each subset the weight vectors $\boldsymbol{w}^{(l)}$ are estimated separately, and thus within each subset the vectors $\pi(\theta_j)$ and the transformed examples $\bar{x}$ have only m components. Consequently, in Equation (13) the inner summation over the examples only runs over the m examples in the subset to which example (x_i, y_i) belongs. Finally, we obtain the parameters of the classifier h by averaging the weight vectors over all subsets, $w_j = \frac{1}{s}\sum_l w_j^{(l)}$.

Mixing the separately learned weight vectors is identical to the mixture weight method by Mann et al. (15) that has been shown, theoretically and empirically, to yield results close to learning a weight vector using all examples at once. Note however that $\boldsymbol{\theta}$ is still learned from the whole, unpartitioned training sample. Using the partitioned estimation for the weight vectors, the overall complexity of the algorithm becomes $O(nm^{1.376} + nmkT)$. For $m \ll n$, the computation becomes tractable even for very large sample sizes n.

4. EMPIRICAL EVALUATION

In this section, we evaluate the proposed algorithm using a large data sample from Yahoo! News United Kingdom[1]. The click log contains browsing session logs, where events from the same user are identified by their browser cookies and sessions are split after 25 minutes of inactivity. We use all sessions from June 2011 for training and the first week of July 2011 for testing. Figure 1 shows the categories, such as *politics/world*, *politics/uk*, *science/space*, in the training set, averaged over the four weeks where different colors correspond to different categories[2].

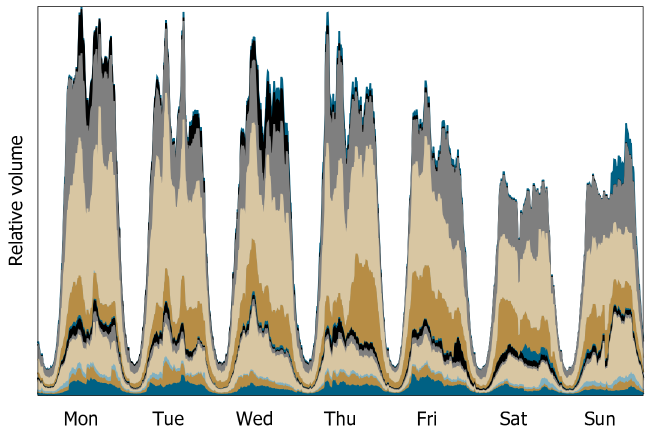

Figure 1: Click volume of categories over time.

The goal of our study is threefold: Firstly, we aim to segment the user sessions of Yahoo! News according to their interests expressed by clicks. Secondly, new sessions need to be classified to one of the segments so that every segment accurately predicts subsequent clicks of the respective user. Finally, the segments need to be interpretable to allow for devising target strategies from the segment description.

[1]All processing is anonymous and aggregated

[2]Colors occur more than once due to the large number of categories.

A typical targeting strategy in our Yahoo! News example could for instance be a dynamic layout of the Web site to advertise news articles of categories that the respective user is probably interested in.

From a data perspective, modeling sequences of clicked categories by Markov processes is straightforward. However, Markov processes, e.g., visualized by transition matrices, are difficult to interpret as the entries encode interests with respect to the previous category. Taking the inferred Markov model properly into account would imply changing the website layout within a session depending on the previous category. A simpler way to obtain interpretable clusters is to use multinomial distributions for the output variables of interest. We use the sequences of user clicks enriched with the respective locations of the clicks. That is, the behavior y consists of the multi-set of subsequently clicked categories c and link sections s. The distribution $P(y|\theta_j)$ is defined as the product of multinomial distributions

$$P(y|\theta) = \prod_l P(c_i|\mu) \prod_j P(s_j|\nu),$$

where μ and ν are the parameter vectors governing the distributions over categories and link sections, respectively.

The attributes x of a session is represented as a binary feature vector encoding the most common referrer domains, the respective category of the first pageview, as well as features encoding the timestamp; we use binary indicators for every day of the week, for each hour of the day, and for each hour of the week. For the collapsed algorithm, we use a linear kernel and randomly partition the training data into disjoint subsets of size 1,000 for computing the predicted log-likelihoods.

4.1 Baselines

We compare the collapsed algorithm with three baselines, the alternating optimization scheme in Section 3.2, a mixture of experts model and a k-means based solution. The *mixture of experts model* (12) minimizes the squared error in terms of the within-cluster log-likelihoods and optimizes the marginal likelihood

$$\sum_i \log \sum_j P(y_i|\theta_j) P(z_i = j|x).$$

Instead of the prior $P(z = j)$ we have a conditional distribution $P(z = j|x)$ which is defined in analogy to the collapsed algorithm as

$$P(z = j|x) \propto \exp(\sum_i \alpha_{j,i} k(x_i, x)).$$

The mixture of experts model is optimized with a standard EM-algorithm and therefore provides only probabilistic cluster assignments and does not take into account that sessions need to be assigned to only a single cluster.

The third baseline is derived from the straightforward, yet somewhat naïve, approach to segment the input space first and only then optimize the generative model in each cluster. The drawback of this non-iterative approach is that it does generally not lead to homogeneous behavior within clusters because the segments are fixed when estimating the generative models. We use k-means for finding the clustering, and estimate segment paramters $\boldsymbol{\theta}$ by maximum likelihood based on the hard cluster assignments. The classifier h classifies a new instance into the cluster with the nearest centroid.

In each setting, every algorithm is deployed 10 times with random parameter initializations and in the remainder we only report the results of the run with highest training likelihood.

4.2 Convergence

In this section, we evaluate the convergence behavior of the collapsed algorithm. Recall that the collapsed algorithm optimizes an approximate objective, where the hard cluster assignments are replaced by a soft-max controlled by an increasing factor ρ. To cancel out effects caused by the approximation, we substitute the resulting $\boldsymbol{\theta}$ into the exact optimization criterion in Equation (2) and measure the respective objective value. Note that the results do not necessarily increase monotonically.

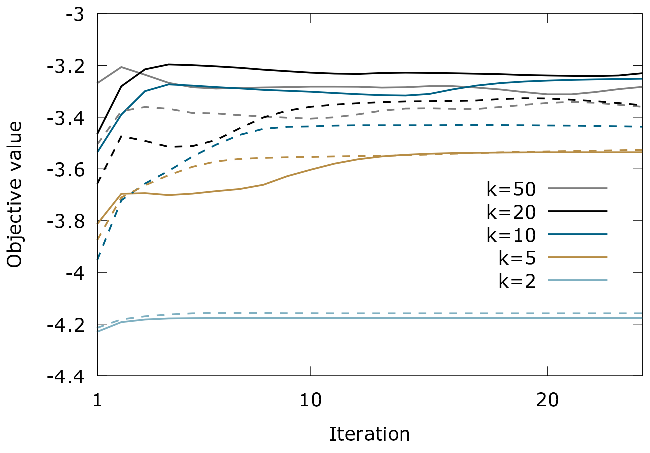

Figure 2: Objective values for the collapsed algorithm (solid) and the mixture of experts baseline (dashed), for different numbers of clusters k.

Figure 2 shows the results for different numbers of clusters for the collapsed algorithm (solid curves). For comparison, we also added the mixture of experts baseline (dashed curves). As expected, the true objective value is not monotonic, since both algorithms optimize an approximation to the exact optimization criterion. The figure also shows that the best values are obtained after at most 20 iterations.

4.3 Predictive Performance

To evaluate the performance of the collapsed algorithm, we measure its predictive accuracy in terms of how well future behavior can be predicted. The classifier and the segmentation are learned jointly as described in Section 3 using the training set and then deployed to the test set. The sessions in the test set are first classified by the classifier in one of the segments which is then used to predict the future clicks of the user. Since the final prediction is a complex variable, we refrain from expressing the performance in terms of error rates and measure the predictive log-likelihood $\log P(y|\theta_{h(x)})$ instead. We compare the collapsed algorithm to the alternating optimization scheme, the mixture of experts model, and the k-means based solution. We report on averages and standard errors over 10 repetitions with different random initializations.

Figure 3 shows the predictive performance for varying numbers of clusters. Not surprisingly, all methods perform

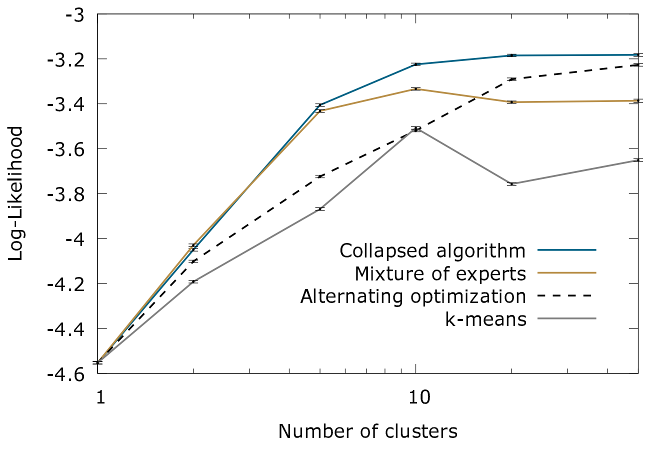

Figure 3: Averaged predictive performance and standard error.

equally worse for only a single cluster. For only a few clusters, the mixture of experts baseline performs about as well as the collapsed algorithm. We credit this finding to the existence of easy-to-reach solutions that do not necessarily require hard cluster assignments in the $\boldsymbol{\theta}$-steps. However, when the number of clusters grows, the performance of the mixture of experts approach decreases slightly while that of the collapsed model increases. Here it becomes more and more important to select the parameters in a way that allows to discriminate well between the clusters, and thus the collapsed algorithm outperforms the baselines significantly. The alternating algorithm and the k-means baseline perform significantly worse than the collapsed algorithm. Only for 20 and more clusters the alternating algorithm produces better results than the mixture of experts model. Note that the k-means performs worst as it does not use an alternating update schema but first learns the clustering and then estimates the generative models using the fixed segments.

It is apparent that the predictive performance levels off after increasing the number of clusters beyond 10. Intuitively, this observation can be explained by a trade-off between classification and segmentation: even if a more fine-grained clustering would be able to predict the future behavior more accurately, the classifier cannot discriminate well between a larger number of similar clusters to identify the best-matching segment. We observe a natural trade-off between predictive power and the effort that has to be spent for developing and maintaining target strategies for a large number of market segments.

The execution time of the collapsed algorithm for a solution with 10 clusters is within the range of 3 hours, compared to about an hour each for the mixture of experts and the k-means baselines. The alternating optimization however takes about 11 hours which renders its application infeasible in practice.

4.4 Discussion

Market segmentation aims at grouping similar individuals of a population together that share the same needs or that have similar demands. The goal is to target individuals within the same segment jointly e.g., to advertise a new product. To this end, the segments need to be interpretable

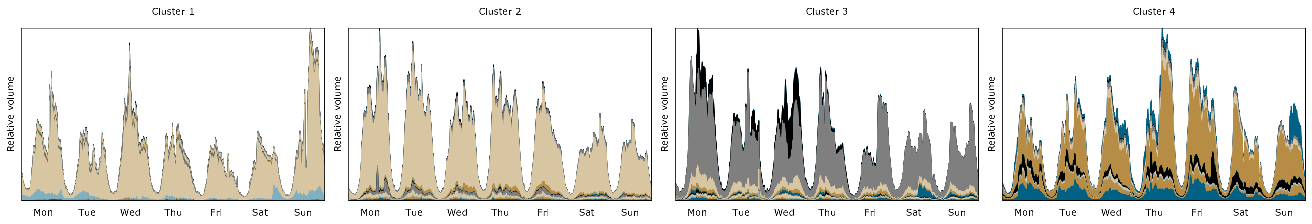

Figure 5: Click volumes of categories over time for the four clusters.

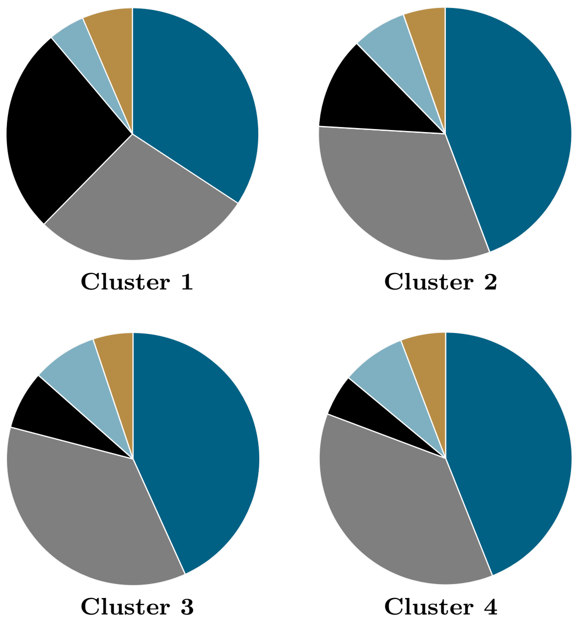

Figure 4: Visualization of click frequencies for the five most frequent link locations using four clusters.

to derive a concise description of the segments that can be converted into a segment-specific targeting strategy.

In our collapsed algorithm, generative models in each segment encode the contained behavior and interest. The flexibility of the probabilistic inference machinery allows us to project the behavior onto discriminative variables to visualize different characteristics of the clusters. In this section we give two examples for such projections to visualize differently distributed user behavior across the clustering. For simplicity, we use a solution with four clusters.

The first example shows a visualization of segment-specific user clicks in terms of their location on the Web page. Including the location of clicks is necessary for altering the layout dynamically as changes in frequently clicked areas will have impact the behavior more than substituting a redundant and less clicked widget. We focus on the five modules of the Web site that receive the highest number of clicks in the data.

Figure 4 shows the results. Segments 2, 3, and 4 exhibit very similar click behavior in terms of the clicked modules. By contrast, cluster 1 differs significantly in the usage of the Web components. On average, users in cluster 1 prefer the location visualized in black over the alternatives compared to users in the other segments. This observation could be exploited to directly devise target strategies. While members of cluster 2–4 should be addressed by changing the content of the modules visualized in gray or dark blue, users in the first segment could also be triggered by the module encoded in black.

Analogously, the behavior could be projected on the categories to visualize the respective distribution of categories for each segment. However, we choose to show a more interesting projection for lack of space. The incorporation of the timestamps of the sessions allows us to visualize the clusters in time. As the feature representation of timestamps encompasses one week, Figure 5 shows the average category distribution across the days of the week where different colors correspond to different categories.[3]

Apparently, the clusters do not only differ in terms of the categories but also specialize on certain periods in time because the segments are optimized using *all* available data, that is, attribute *and* behavior encoding variables. The first cluster clearly specializes on Sundays and is characterized by a clean topic distribution. The three other cluster also possess dominant categories but focuses more on working days than on weekends. Cluster 4 contains the most diverse set of categories and acts like a basin for categories that are not as easy to discriminate. Here it becomes obvious that a solution with only four clusters may not be optimal for the task at hand. When we increase the maximal number clusters, the category distribution of clusters becomes cleaner that is less categories are likely. Additionally, clusters adapt better to specialized periods such as working days or weekends for larger k.

Taking various such projections into account describes segments from different angles and helps to find a concise targeting strategy. For instance, knowing the articles that are likely to be read in an ongoing session helps to address the respective user in various ways including displaying ads. Incorporating context informations such as the click behavior of the segments, finally allows for tailoring web pages to each segment and to increase the overall user experience.

5. CONCLUSION

We studied discriminative clustering for structured and complex response variables that can be represented as generative models. The problem setting matches market segmentation tasks where populations are to be segmented into disjoint groups. Solving market segmentation-like problems appropriately not only involves a clustering of the individuals but also learning a classifier that discriminates well be-

[3]Colors are again reused due to the large number of categories.

tween the segments, for instance to allow for classifying new customers to one of the groups. The two components need to be learned jointly and have access to different pieces of information about the individuals: the classifier needs to group individuals on the basis of *a priori* available informations while the clustering aims at grouping people with similar (future) needs or behavior.

We devised two algorithms based on alternating optimization and collapsed inference, respectively. Empirical results showed that the collapsed variant is not only more efficient but also predicts accurately the click behavior of users for Yahoo! News. The generative nature of the clustering led to interpretable clusters. We showed how projections of the clustering on only a few variables allowed for targeting the detected segments individually and contributed to user understanding.

Our approach is not restricted to Yahoo! News and can generally be applied to arbitrary market segmentation tasks and other Web sites to improve the overall user experience. As our approach is orthogonal to personalized approaches, future work will study the integration of both frameworks.

Acknowledgements

Part of this work was supported by the German Science Foundation (DFG) under the reference number GA 1615/1-1. This research is partially supported by European Community's Seventh Framework Programme FP7/2007-2013 under the ARCOMEM and SOCIAL SENSOR projects, by the Spanish Centre for the Development of Industrial Technology under the CENIT program, project CEN-20101037 (www.cenitsocialmedia.es), "Social Media.", and by Grant TIN2009-14560-C03-01 of the Ministry of Science and Innovation of Spain.

References

[1] N. Bansal, A. Blum, and S. Chawla. Correlation clustering. *Machine Learning*, 56(1-3):89–113, 2004.

[2] S. Bickel and T. Scheffer. Multi-view clustering. In *ICDM 2004*. Citeseer, 2004.

[3] U. Brefeld, P. Geibel, and F. Wysotzki. Support vector machines with example dependent costs. In *ECML*, 2003.

[4] R. D'Andrade. U-statistic hierarchical clustering. *Psychometrika*, 4:58–67, 1978.

[5] F. De la Torre and T. Kanade. Discriminative cluster analysis. In *ICML 2006*, pages 241–248. ACM, 2006.

[6] A. Dempster, N. Laird, and D. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 1–38, 1977.

[7] P. D'Urso and L. D. Giovanni. Temporal self-organizing maps for telecommunications market segmentation. *Neurocomput.*, 71:2880–2892, 2008.

[8] R. Gomes, A. Krause, and P. Perona. Discriminative clustering by regularized information maximization. In *Advances in Neural Information Processing Systems*, 2010.

[9] J. Huang, G. Tzeng, and C. Ong. Marketing segmentation using support vector clustering. *Expert systems with applications*, 32(2):313–317, 2007.

[10] R. Jacobs, M. Jordan, S. Nowlan, and G. Hinton. Adaptive mixtures of local experts. *Neural computation*, 3(1):79–87, 1991.

[11] S. C. Johnson. Hierarchical clustering schemes. *Psychometrika*, 2:241–254, 1967.

[12] M. Jordan and R. Jacobs. Hierarchical mixtures of experts and the em algorithm. *Neural computation*, 6(2):181–214, 1994.

[13] M. Y. Kiang, M. Y. Hu, and D. M. Fisher. An extended self-organizing map network for market segmentation – a telecommunication example. *Decision Support Systems*, 42:36–47, 2006.

[14] S. P. Lloyd. Least square quantization in pcm. *IEEE Transactions on Information Theory*, 28(2):129–137, 1982.

[15] G. Mann, R. McDonald, M. Mohri, N. Silberman, and D. Walker. Efficient large-scale distributed training of conditional maximum entropy models. *Advances in Neural Information Processing Systems*, 22:1231–1239, 2009.

[16] M. Namvar, M. Gholamian, and S. KhakAbi. A two phase clustering method for intelligent customer segmentation. In *2010 International Conference on Intelligent Systems, Modelling and Simulation*, pages 215–219. IEEE, 2010.

[17] J. Sinkkonen, S. Kaski, and J. Nikkilä. Discriminative clustering: Optimal contingency tables by learning metrics. *Machine Learning: ECML 2002*, pages 109–137, 2002.

[18] K. Wagstaff, C. Cardie, S. Rogers, and S. Schrödl. Constrained k-means clustering with background knowledge. In *ICML*, 2001.

[19] M. Wedel and W. Kamakura. *Market segmentation: conceptual and methodological foundations*, volume 8. Springer, 2000.

[20] L. Xu, J. Neufeld, B. Larson, and D. Schuurmans. Maximum margin clustering. *Advances in neural information processing systems*, 17:1537–1544, 2005.

[21] J. Ye, Z. Zhao, and M. Wu. Discriminative k-means for clustering. In *Advances in Neural Information Processing Systems*, 2007.

[22] W. Yu and G. Qiang. Customer segmentation of port based on the multi-instance kernel k-aggregate clustering algorithm. In *ICMSE 2007*, pages 210–215, 2007.

[23] B. Zhao, J. Kwok, and C. Zhang. Maximum margin clustering with multivariate loss function. In *ICDM'09*, pages 637–646. IEEE, 2009.

[24] B. Zhao, F. Wang, and C. Zhang. Efficient multiclass maximum margin clustering. In *ICML 2008*, pages 1248–1255. ACM, 2008.