Partitioning of Image Datasets using Discriminative Context Information

Christoph H. Lampert
Max-Planck-Institute for Biological Cybernetics, Tübingen, Germany
chl@tuebingen.mpg.de

Abstract

We propose a new method to partition an unlabeled dataset, called Discriminative Context Partitioning (DCP)*. It is motivated by the idea of splitting the dataset based only on how well the resulting parts can be separated from a context class of disjoint data points. This is in contrast to typical clustering techniques like* K-means *that are based on a generative model by implicitly or explicitly searching for modes in the distribution of samples.*

The discriminative criterion in DCP *avoids the problems that density based methods have when the a priori assumption of multimodality is violated, when the number of samples becomes small in relation to the dimensionality of the feature space, or if the cluster sizes are strongly unbalanced. We formulate* DCP*'s separation property as a large-margin criterion, and show how the resulting optimization problem can be solved efficiently. Experiments on the MNIST and USPS datasets of handwritten digits and on a subset of the Caltech256 dataset show that, given a suitable context,* DCP *can achieve good results even in situation where density-based clustering techniques fail.*

1. Introduction

The question how to split a collection of data into meaningful subgroups is one of the oldest and most difficult problems in the area of machine learning. In the absence of additional information, such as training labels or a quality function, it is even ill-posed. In practice, it depends on the applications and thereby ultimately on the human, which partitionings of the data are preferred.

When building an algorithm one needs to formulate a computable criterion for which partitions are good. Most existing methods, such as *K-means*, *mean shift* or *spectral clustering*, rely on a criterion that can be formalized as the *Cluster Assumption* [7, 15]. It states that samples belong to the same cluster, if they lie within a common region of high sample density, whereas they belong to different clusters, if they are separated by a region of low sample density.

However, not all realistic data is of that type, and humans have many other ways to split a dataset. This is particularly apparent for image collections. Figure 1 shows a simple example of handwritten digits. It consists of an arrangement of handwritten digits in a 2-dimensional projection obtained by PCA. Humans can split this set easily, *e.g.* identifying the most discriminative, extremal examples of the left- and the right-slanted digits with a transitional region of upright digits in between. As we will see later, density based clustering method fail to detect this. One can come up with many other examples of this effect: a set of *car* images can be split into *front-views*, *side-view* and *rear-view* examples, even if many images do not fall perfectly into any of the categories. Obviously, such a set does not fulfill the cluster assumption, as the transition area between poses might be densely populated as well. Such distributions are typical for generating processes that involve continuous latent factors like *size*, *color* or *perspective*.

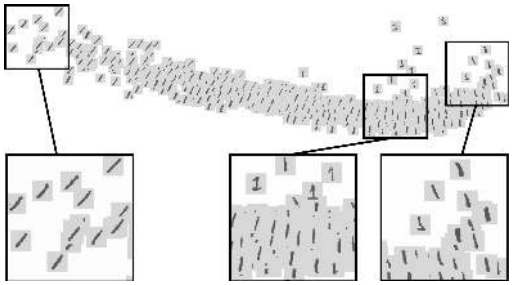


Figure 1. A 2D embedding of the digit '1' image in the MNIST dataset. Humans easily identify a large region of *right-slanted* digits (left) and a smaller *left-slanted* region (right). The parts are connected by a densely populated transitional region.

In this paper we propose *discriminative context partioning (DCP)*, which allows splitting of image datasets like in Figure 1 into subparts without applying the cluster assumption. Instead, *DCP* relies in a purely discriminative way on how well the parts can be separated from a disjoint sample set by a generalized large-margin criterion. The additional samples act as a *rest of the world* class, that can be interpreted as an analog to the context and background knowledge that humans have.

Before we explain the details of *DCP*, we give an

overview of related work in the field of clustering and of previous uses of context sets in machine learning. Here and in the following, we will call the generic process of splitting a dataset into parts *partitioning*. We reserve the word *clustering* for methods where the resulting parts are *clusters* in the density-based sense that the word is most commonly used in. *DCP* is not a clustering method by this definition.

2. Related Work

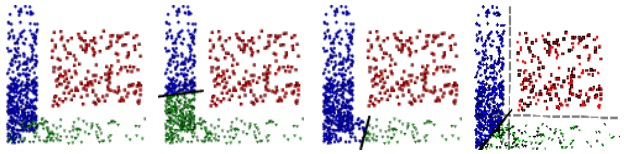
There is a large amount of research on unsupervised, supervised and semi-supervised learning that is related to the problem of splitting a dataset into parts. In the following, we will only look at methods that provide a hard assignment of samples to parts, because this is also the way *DCP* works. We sort the method by how the resulting parts are formed and represented.

Prototype-based Clustering. A common way of representation is to choose one prototype per cluster. Each data point is assigned by a nearest-neighbor classifier to the cluster of the prototype it lies closest to. The prototypes are typically formed as linear combinations of sample vectors, as in the many variants of the classical K-*means clustering* [13]—which is by far the most popular clustering method today—or in *mean-shift clustering* [8, 9]. Alternatively, a selection of the samples themselves can serve as prototypes, as in K-*median clustering* and *median-shifts* [18]. As a variant, *spectral clustering* [14, 19] has recently gained popularity, because it combines the flexibility of an arbitrary distance measure with the simplicity of the K-means search for cluster prototypes. Generally, prototype-based methods perform a form of *learned vector quantization*, and the resulting clusters typically correspond to maxima in the sample density, possibly after a preprocessing step.

Graph-based Clustering. Graph-based clustering techniques represent a data set as a graph, and partitions non parametrically as subgraphs. This makes them applicable also in situations where the samples do not lie in a vector space. Classical graph-based methods work hierarchically as *agglomerative clustering*, or as *divisive clustering* [12]. Both approaches favor cluster boundaries between samples that have a large distances from each other.

Classifier-based Clustering. Instead of representing the clusters in terms of their elements, classifier-based clustering methods form parametric partitions of the whole input space. This idea is *e.g*. employed by *maximum-margin clustering* [24] which uses large-margin classifiers to identify those parts of the dataset that can be well separated from each other. Alternatively, in *support vector clustering* [4], the contour lines of a trained one-class SVM with Gaussian kernel form the cluster boundaries.

The use of large-margin classifiers makes these approaches related to *DCP*, but a fundamental difference is, that all presented approaches rely in some way on the *cluster assumption* to identify cluster boundaries, which *DCP* does not.



(a) ground truth (b) K-means (c) spectral clst. (d) DCP

Figure 2. Example dataset consisting of 2 overlapping parts ($\bigcirc$,$\bigtriangledown$) in an L-shaped arrangement. *DCP* with access to a *context class* ($\square$) identifies the intended parts. K-means and spectral clustering without context yield worse results.

Inclusion of Side Information. Research on the inclusion of side-information into clustering has mainly concentrated on adding low level constraints to K-means-like algorithms, *e.g*. by bounding the absolute or relative size of clusters [5]. Another common approach is to use weak label constraints, such as sample pairs known to have *the same* or *different* labels [22].

To our knowledge, there is no published work on partitioning by only using a *context* or *rest-of-the-world* class, but context information has frequently been used in *semi-supervised* learning. The *Transductive Support Vector Machine (TSVM)* [21] makes use of labeled and unlabeled data from the same distribution by constructing a decision margin that separates the labeled samples but has few unlabeled samples falling into it. Alternatively, *learning with a Universum* [23] makes use of additional data that belongs to a *different class* than the training data. It relies on constructing a classifier that separates the labeled data but is *maximally indecisive* about the context data.

Apart from not requiring any label information, *DCP*'s way of using context also differs from both the TSVM's and the *Universum*'s approach. *DCP* does not rely on how the context class can be split, but the context serves solely as a stationary reference set. Only the dataset itself is split by separating each of the parts individually from the context.

3. Discriminative Context Partitioning

We introduce the rationale behind the *DCP* algorithm on the example of Figure 2, which was created by applying *DCP* and other clustering methods to a toy dataset in $\mathbb{R}^2$. It contains an L-shaped dataset that we would like to split into the vertical (blue) and horizontal (green) bars, as would be a natural choice to a human. Additionally, a disjoint rectangular context class (red) is available, that implicitly encodes the geometry of our intended split.

Figure 2(b) shows that the K-means algorithm does not find the intended split, presumably because the sample distribution is not multimodal, and the K-means objective

tends to favor equally sized clusters in such cases. Clustering methods based on local distances, *e.g.* spectral clustering with a Gaussian distance function, split the dataset in a low density region (Figure 2(c)) which in our case is not the intended split, either.

DCP works fundamentally different. It tries to find that partitioning of the data, in which each part individually can be separated better from the context class than the combined data set, where we measure the separation by the margin achieved by an SVM-like large margin classifier. The decision boundaries of the best achievable classifiers are shown in Figure 2(d) as dashed lines. The resulting split of the L is very close to the ground truth.

By using the SVM margin as a discriminative measure of separation, we do not require assumptions about the densities of the parts. It is only their global shape that influences the class decision. This makes *DCP* practically independent of how densely the parts are sampled. The boundary, at which the dataset is split, is not modelled directly, but arises from the orientation of the classifiers. The method is therefore indifferent to whether it splits the dataset in a high-density or in a low-density region.

Similar ideas of splitting a dataset into more discriminative subparts has been employed earlier *e.g.* in the *mixture-of-experts* framework [11]. A construction similar to *DCP* has been proposed in a supervised setting as a *multi-prototype SVM* [1]. However, *DCP* differs from these as it does not require training labels and therefore also does not aim at improving the classification accuracy. Instead, we are interested in the most discriminative split itself.

A characteristic property of *DCP* is, that the choice of the context class mainly determines how the data set is going to be split. Since the only constraint there is, that the context is not a part of the dataset, usually more than one possible context class is available, and they yield different partitioning. This effect can be used advantageously, allowing a user to explore different splits of the data. In contrast to forming random splits, the context-driven exploration respects the global geometry of the data.

Already from the simple situation in Figure 2 we can see that *DCP* differs from other techniques to partition a dataset, in particular from classical *clustering* techniques, in three main aspects:

- It allows the partitioning of dataset that do not adhere the clustering assumption, *e.g. unimodal datasets*.
- It is *insensitive* to how large the parts are and how densely they are sampled.
- By choosing different context classes, the data can be split with respect to different aspects, giving a human user an *interactive and explorative* way to analyze a dataset.

We will present supporting experiments for these claims in Section 5.

3.1. Mathematical Formulation

To clarify the informal characterization in the previous section, we now give the mathematical definition of *discriminant context partitioning* that allows us to come up with an efficient algorithm. For this, we assume fixed finite datasets X and Z. X is to be split and Z is the disjoint context. For a fixed $K \in \mathbb{N}$, we are interested in the *most discriminative split* of X into K parts with respect to Z, according to the following definition.

Definition 1. *Let $\mathcal{C}$ be a set of classifier-like functions such that for any $c \in \mathcal{C}$ and any subset $A \subset X$ we have a measure $sep_f(A, Z)$ of how well f discriminates between A and Z. Let $X_1 \cup \cdots \cup X_K = X$ be a decomposition of X. Then the* total separation *of this split is*

$$sep(X_1, \ldots, X_K \,;\, Z) := \sum\nolimits_{k=1}^{K} \max_{f \in \mathcal{C}} \; sep_f(X_k, Z). \quad (1)$$

*A decomposition $X_1^* \cup \cdots \cup X_K^* = X$ is called a **most discriminative K-split of X with respect to Z**, if it maximizes the total separation over all possible decompositions of X.*

Different choices for $\mathcal{C}$ are possible. To make the definition useful, the separation measure has to take into account global, *e.g.* geometric, properties of the sets. For example, defining the separation between sets as the minimum distance between elements in some metric space will not give interesting properties to the most discriminative split, because too large parts of X can end up in either subset without influencing the separation score.

In a framework of linear hard-margin classifiers, however, a natural choice for $sep_f(A, Z)$ would be the margin by which the classifier f separates A and Z. In fact, the resulting splits could then look similar to DCP's results in Figure 2(d). However, this measure is undefined if A and Z are not linearly separable.

For the *DCP* algorithm, we choose $\mathcal{C}$ as a set of soft-margin generalized linear classifiers, *e.g.* soft-margin SVMs with arbitrary kernel. The separation between sets is measured by the negative of the SVM objective function

$$sep_f(X, Z) := -\frac{1}{2}\|f\|_{\mathcal{H}}^2 - \sum_{x \in X} \ell(1 - f(x)) - \sum_{z \in Z} \ell(1 + f(z)), \quad (2)$$

where ℓ is a monotonous convex loss function that penalizes margin violations, *e.g.* the hinge loss or the quadratic loss. This choice allows us to intuitively understand that a large separation between two sets still corresponds to a large margin between them. The optimal f for given A and Z can be computed efficiently by the usual SVM training procedure.

A solution to the discrete partitioning problem is now given by the following proposition. For the sake of readability we only formulate and prove it for linear classifiers $f(x) := \langle w, x\rangle$ here. The non-linear case is obtained by kernelization in Section 4.

Proposition 1. *The most* discriminative partitioning *of X with respect to Z is given by*

$$X_k^* := \{\, x \in X \;:\; \operatorname*{argmax}_{k'=1,\dots,K} \langle w_{k'}^*, x\rangle = k \,\}, \qquad (3)$$

for $k = 1, \dots, K$, where $(w_k^*)_{k=1,\dots,K}$ minimizes

$$J(w_1, \dots, w_K) = \frac{1}{2}\sum_{k=1}^{K} ||w_k||^2 + \sum_{k=1}^{K}\sum_{z\in Z} \ell(1 + \langle w_k, z\rangle) + \sum_{x\in X} \ell(1 - \max_k \langle w_k, x\rangle). \qquad (4)$$

Here and in the following, the argmax should be interpreted in an exclusive way, such that each sample $x \in X$ cannot be assigned to more than one partition X_k^*. We have also adopted the convention that the linear classifiers are written without a bias term, but that the samples are augmented in a way where one component of w_k takes this role.

Proof. Because of the way the X_k^* are constructed, it is clear that their union is all of X. We therefore only have to show that $(X_k^*)_{k=1,\dots,K}$ maximizes the total separation as given in Definition 1.

First, we parameterize the set of all possible splits by introducing a binary membership matrix $\boldsymbol{\lambda} = (\lambda_k^i)_{x_i\in X, k=1,\dots,K}$ for the elements of X, where $\lambda_k^i = 1$ if and only if x_i is an element of X_k. We can ensure that $X = \bigcup_k X^*$ by demanding that $\sum_k \lambda_k^i = 1$. Every $\boldsymbol{\lambda}$ now uniquely represented a split of X by the relation $X_k = \{x_i \in X : \lambda_k^i = 1\}$.

Note that by imposing additional constraints on $\boldsymbol{\lambda}$, we could also incorporate other side information like known cluster membership for some samples, samples sharing or not sharing their labels, or limits on the cluster sizes.

To prove Proposition 1, we will first study the situation where the partitioning matrix $\boldsymbol{\lambda}$ is fixed, and afterwards the maximization over all these cases.

Large margin classification with fixed partitions: Inserting Equation (2) with a linear $f_k(x) = \langle w_k, x\rangle$ into Equation (1) we obtain

$$sep(X_1, \dots, X_K; Z) = -\min_{\boldsymbol{w}} \sum_{k=1}^{K} \Big\{ \frac{1}{2}||w_k||^2 + \sum_{z\in Z} \ell(1 + \langle w_k, z\rangle) + \sum_{x_i\in X} \lambda_k^i \ell(1 - \langle w_k, x^i\rangle) \Big\}, \qquad (5)$$

where $\boldsymbol{w} = (w_k)_k$. Since we assumed λ_k^i fixed, and the components for different K do not interact with each other, the optimal $\boldsymbol{w}$ can be found by training K independent soft-margin SVMs with slack penalization ℓ, where each SVM uses all of Z as negative examples but only those $x_i \in X$ as positive example for which $\lambda_k^i = 1$.

Joint minimization over partitions and classifiers: Finding the most discriminative partition corresponds to maximizing Equation (5) over $\boldsymbol{\lambda}$ or equivalently minimizing its negative:

$$\boldsymbol{\lambda}^* = \operatorname*{argmin}_{\boldsymbol{\lambda}} \min_{\boldsymbol{w}} \sum_{k=1}^{K} \Big\{ \frac{1}{2}||w_k||^2 + \sum_{z\in Z} \ell(1 + \langle w_k, z\rangle) + \sum_{x_i\in X} \lambda_k^i \ell(1 - \langle w_k, x^i\rangle) \Big\}, \qquad (6)$$

subject to $\lambda_k^i \in \{0, 1\}$, $\sum_{k=1}^K \lambda_k^i = 1$ for all $i = 1, \dots, N$. Note that in a minimum $\operatorname{argmin}_{\boldsymbol{\lambda}}$, we will have

$$\lambda_k^{*i} \neq 0 \quad \text{only if} \quad \ell(1 - \langle w_k, x^i\rangle) = \min_{k'} \ell(1 - \langle w_{k'}, x^i\rangle), \qquad (7)$$

because otherwise we could increase the separation by decreasing λ_k^{*i} while at the same time increasing the minimal $\lambda_{k'}^{*i}$. Using the monotonicity of ℓ, we obtain that it suffices to solve the unconstrained optimization problem

$$\boldsymbol{w}^* = \operatorname*{argmin}_{\boldsymbol{w}} \frac{1}{2}\sum_{k=1}^{K} ||w_k||^2 + \sum_{z\in Z}\sum_{k=1}^{K} \ell(1 + \langle w_k, z\rangle) + \sum_{x\in X} \ell(1 - \max_{k=1,\dots,K} \langle w_k, x\rangle). \qquad (8)$$

The corresponding best partition $X_k^* = X_k^{\boldsymbol{\lambda}}$ can now be reconstructed by going backwards through the calculation and undoing the simplifications, yielding

$$\begin{aligned} X_k^{\boldsymbol{\lambda}} &= \{x_i \in X : \lambda_k^{*i} = 1\} \\ &= \{x \in X : \ell(1 - \langle w_k^*, x\rangle) = \min_{k'} \ell(1 - \langle w_{k'}^*, x\rangle)\} \\ &= \{x \in X : k = \operatorname*{argmax}_{k'} \langle w_{k'}^*, x\rangle\} \end{aligned} \qquad (9)$$

which concludes the proof. □

3.2. Numerical optimization

The previous section gave us two equivalent formulations to find the most discriminative partition. Minimizing either Equation (6) or Equation (8) will yield the desired split. However, neither of the formulations is convex, and we therefore cannot expect to easily find a globally optimal solution as in the case of fixed partitions. In the following, we analyze the properties of both minimization problem, showing that they give rise to two different implementation for *DCP*.

3.2.1 Optimization by Deterministic Annealing

Equation (6) is an example of a non-linear mixed-integer program. In general, these are difficult so solve because of the combinatoric component. However, minimization problems similar to (6) occur *e.g.* in the training step of the *Transductive Support Vector Machine (TSVM)*, and efficient optimization techniques have been developed, *e.g.* based on *Deterministic Annealing* [20].

Deterministic Annealing treats the entries of $\boldsymbol{\lambda}$ as random variables following a probability distribution $p(\boldsymbol{\lambda})$. Studying the expectation value of the objective function over $p(\boldsymbol{\lambda})$, we obtain a homotopy method for solving (6) that iterates the following steps: for a fixed entropy level H we calculate the minimizing distribution $p(\boldsymbol{\lambda})$ while keeping $\boldsymbol{w}$ fixed. This is easy if we allow only factorizing distributions since $\boldsymbol{\lambda}$ is binary. Fixing this $p(\boldsymbol{\lambda})$, we then minimize over $\boldsymbol{w}$, which is a convex problem. By varying H, we obtain a sequence of simpler problems that converges smoothly to the original non-convex and combinatoric problem for $H \to 0$.

Applying this technique to (6) results in an algorithm that is similar to the *Stochastic MProtSVM Optimization* method proposed in [1] for training a multi-prototype SVM.

3.2.2 Optimization by Gradient Descent in the Primal

In deriving Equation (8), we were able to remove the combinatoric component, but with the negative side effect that the resulting $\max$-term makes the objective function non-convex in $\boldsymbol{w}$. Nevertheless, the resulting form is very similar to a normal SVM optimization in primal form [6]. As proposed there, we can apply a gradient descent based technique, either relying on subgradients [3] or smoothing the objective function first, *e.g.* by a $\mathrm{softmax}$ operation. The same applies to the loss function, where we can approximate *e.g.* the non-differentiable hinge loss by the Huber loss. As always when relying on gradient descent for a non-convex problem, convergence to a global optimum is not guaranteed, but we have to find a good start vector to not get trapped in a local minimum. Our experiments show that a random initialization for $\boldsymbol{w}$ is sufficient if we perform 10 or 20 restarts and choose the solution of maximal separation score. Alternatively, one can start the process with vectors $\boldsymbol{w}$ that were obtained my solving Equation (5) given some suboptimal partitioning of the space, *e.g.* obtained by K-means clustering.

4. Kernelization

So far we have only studied the case of linear classifiers. As usually for SVM-like algorithms, we can make use of the kernel trick [2] to extend our method to the non-linear case. The use of kernels is also necessary in situations where linear operations on the samples are not defined, *e.g.* to find splits of non-vectorial data. Since we have not represented our optimization problem in the dual, where kernelization is a standard procedure, we give a sketch how the kernel trick can be applied also in the primal, following the description in [16].

Let $\mathcal{C}$ be the space of soft-margin SVMs with kernel function $\mathcal{K}(.,.)$ that gives rise to non-linear classifiers f in a reproducing kernel Hilbert space $\mathcal{H}$. Instead of Equation (4), the separation measure becomes

$$\frac{1}{2}\sum_{k=1}^{K}||f_k||^2_{\mathcal{H}} + \sum_{z\in Z}\sum_{k=1}^{K}\ell(1+f_k(z)) + \sum_{x\in X}\ell(1-\max_k f_k(x)), \tag{10}$$

which we have to minimize over all tuples $(f_k)_k \in \mathcal{H}^K$. From this we can derive $X_k^* := \{x \in X : \mathrm{argmin}_{k'} f_{k'}(x) = k\}$ as a most discriminative split of X. We apply the representer theorem [17] and obtain $f_k = \sum_j \beta_j^k \mathcal{K}(.,y^j)$ for suitable coefficients $\boldsymbol{\beta} = (\beta_j^k)_{j,k}$ and $y^j \in X \cup Z$. The objective function becomes

$$\begin{aligned} J_{\mathcal{K}}(\boldsymbol{\beta}) = &\frac{1}{2}\sum_{k=1}^{K}\sum_{y^i,y^j\in X\cup Z}\beta_i^k\beta_j^k\mathcal{K}(y^i,y^j) \\ &+ \sum_{z\in Z}\sum_{k=1}^{K}\ell\Big(1+\sum_{y^j\in X\cup Z}\beta_j^k\mathcal{K}(z,y^j)\Big) \\ &+ \sum_{x\in X}\ell\Big(1-\max_k\sum_{y^j\in X\cup Z}\beta_j^k\mathcal{K}(x,y^j)\Big), \end{aligned} \tag{11}$$

which we have to minimize over $\boldsymbol{\beta} \in \mathbb{R}^{N\times K}$, where $N = |X| + |Z|$ is the total number of samples.

The kernelization runs completely along the lines of other scalar-product based algorithms, but one has to spend some care in the choice of kernels: our objective is not to classify optimally between X and Z, but to identify meaningful structure within X. Z acts only as a tool that encodes information about the feature space in some discriminative way. Therefore, the kernel should not be localized, as *e.g.* RBF-kernels, but has to retain the global geometry and latent structure that is present in X and Z.

5. Experiments

We tested out method on three image datasets, two of handwritten digits and one of generic object categories. All experiments were performed with the gradient-descent approach of Section 3.2.2 from 20 random starting positions using the hinge loss $\ell(x) = C\,[1-x]_+$ with $C = 100$. Other values for C or a higher number of restarts did not have a significant effect on the results.

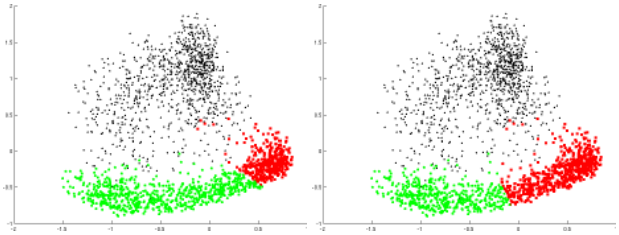


Figure 3. *DCP* (left) and *K-Means* (right) splits of the MNIST digit '1' (bottom) using the digit '7' (top) class as *context*. The K-Means objective favors balanced clusters. *DCP* finds parts which depend on the overall to global geometric situation.

5.1. MNIST

We start by showing how *DCP* handles the situation of datasets not fulfilling the *cluster assumption*, *i.e.* where the parts that we would like to identify are not regions of high sample density separated by a region of low density.

For this, we return to the example of Figure 1 in Section 1. We project the *MNIST* dataset[1] of handwritten digits into $\mathbb{R}^2$ using PCA. The task is to split the set of digits '1' into two parts and for this, we apply K-means and *DCP*, where *DCP* is given access to the digit '7' set as context.

The results are shown in Figure 3 as point clouds: the class 1 is the handle shaped cloud in the bottom, this is the same as Figure 1. The other, more spread out cloud in the top is the set of digits 7. *DCP* splits the set of 1s far to the right, because this yields the best large-margin separation from the context 7. By consulting Figure 1 we see that the split falls into the region of upright digits. Therefore, the split that *DCP* finds discriminates well between the regions of different *slant*. This is also visible from Figure 4, which contains samples from the resulting parts. The smaller partition contains the left slanted and the larger partition contains the right slanted digits. Uprights and irregular shaped 1s lie in the transitional region and occur in both partitions.

In contrast, K-means converges to a split where both clusters basically contain half of the samples, a typical behavior if there are no clear local maxima in the data distribution. The resulting cut falls into the region of right slanted digits, and the clusters do not show an interpretable pattern.

5.2. USPS

In another experiment, we analyze a situation where the *cluster assumption* holds, but which nevertheless shows drawbacks of the inherently generative clustering algorithms. We used the *USPS* database[2] of handwritten digit images, treating each an 8×8 digit image as a vector in $\mathbb{R}^{256}$. We created pairwise mixtures of all combinations of digits, and used K-means as well as *DCP* to separate these again. In another experiment, we used the same setup but mixed all samples of one digit class with only 10% of the other digit class, creating datasets in which the target clusters are strongly unbalanced. With the dataset consisting of mixtures of 2 digits each, there are always 8 disjoint sets of digits remaining, that we used as context for *DCP*.

[1] http://yann.lecun.com/exdb/mnist/index.html
[2] http://www.cs.toronto.edu/~roweis/data.html

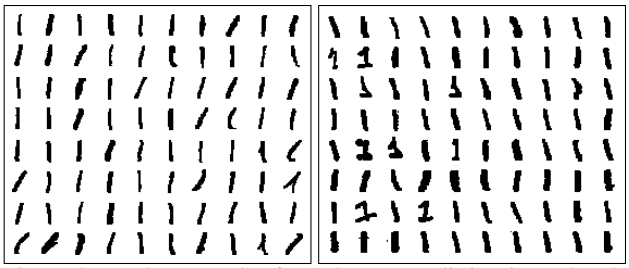
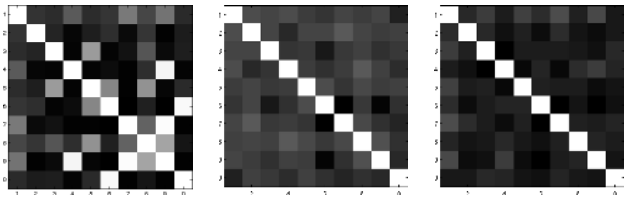
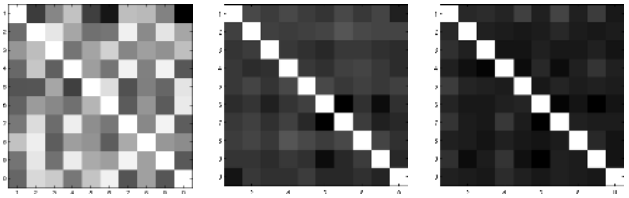


Figure 4. Random samples from the *DCP* split in Figure 3. The left cluster contains mainly right-slanted digits, the right cluster mainly left-slanted and irregularly shaped digits.



(a) balanced mixture: KM (left), DCP-avg (center), DCP-best (right).



(b) unbalanced mixture: KM (left), DCP-avg (center), DCP-best (right).

Figure 5. Errors rate in unmixing USPS digits for K-means (KM) clustering, *DCP* with best context class (DCP-best) and average of *DCP* over all context classes (DCT-avg). Black indicates 0% and White 50% error rate.

Figure 5 visualizes the results for all pairs of digit mixtures in the balanced and unbalanced case. The error rate is indicated by shading, with black meaning 0% and white the maximal possible 50%. The matrices show the result of K-means, of *DCP* averaged over all context classes, and of *DCP* for the best context class in each case. One can see that except for some outliers, K-means manages to separate the balanced mixtures well. This is to be expected, as the K-means objective is well suited for the case of multimodal mixture densities. However, when the mixture becomes unbalanced K-means fails almost always. Presumedly, this happens because the local density maximum of the smaller mixture component disappears.

DCP's averaged performance is comparable with K-means in the balanced situation, except that the scores are more uniformly distributed. However, *DCP* is not affected

	6 vs. 0		2 vs. 0	
	1 : 1	1 : 10	1 : 1	1 : 10
KM	48.8±0.0	46.8±0.4	5.9±0.0	36.3±0.6
DCP-*1*	14.5±3.5	14.2±2.6	22.6±2.1	22.5±2.6
DCP-*2*	27.3±2.0	25.2±2.3	—	—
DCP-*3*	13.8±2.4	14.6±3.6	23.4±1.7	27.5±2.1
DCP-*4*	15.5±2.8	14.0±3.4	13.9±3.2	12.7±2.1
DCP-*5*	21.1±1.9	16.1±3.5	20.4±2.1	19.5±2.2
DCP-*6*	—	—	21.1±1.4	20.7±2.1
DCP-*7*	**3.6±0.9**	**4.5±2.0**	**5.8±2.6**	**6.0±1.8**
DCP-*8*	14.5±3.5	15.1±1.9	22.0±3.1	23.9±3.4
DCP-*9*	6.1±1.9	7.6±3.0	8.0±2.9	6.6±2.5
DCP-*0*	—	—	—	—
avg.	14.6±2.4	13.9±2.8	17.2±2.4	17.4±2.4

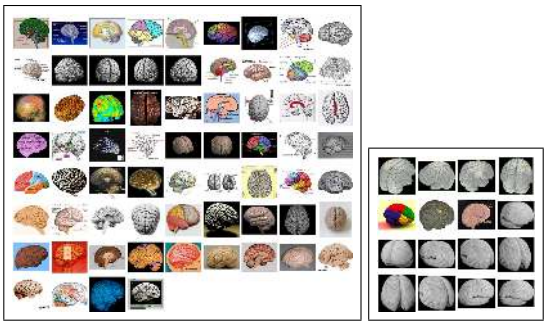
Table 1. Unsupervised separation of USPS digits by K-means (KM) and *DCP* with context classes 0–9 (DCP-n): The column titles indicate the digit pair that has been mixed and the mixing ratio. The reported numbers are average error rates [%] with 95% confidence intervals from 100 runs. See Section 5.2 for details.

by the mixture ratio, thereby clearly outperforming K-means in the unbalanced case. For all mixtures, at least one of the eight context classes causes a very good split, typically better than the one by K-means. One can see this from the minimal *DCP* error rates, which are generally lower than the average ones. However, as we have mentioned in the introduction, there is no *a priori* best context class. In a real application, a decision such as which context yields the best results requires a human or a separate automatic system to judge the splits.

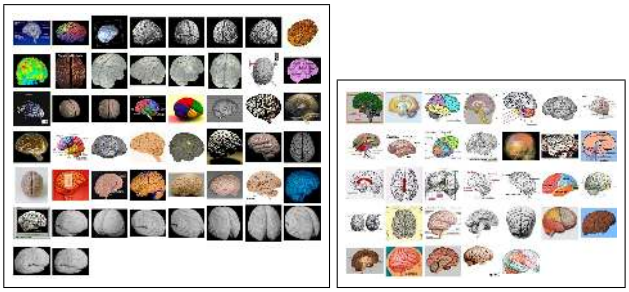
Table 1 shows the effect of the different context classes for two of the mixtures. '6' vs. '0' is a pair that K-means fails to separate. *DCP*'s score clearly depends on the context used, even though it is always better than the K-means solution. Two of the context classes perform better than the others, with the best one achieving excellent results of below 5% error. This is consistent for the balanced as well as for the unbalanced mixture. The second mixture, '2' vs. '0' is one that K-means manages to solve well, better than the average performance of *DCP* and on par with *DCP* with best context class. However, when the data becomes unbalanced, K-means' error rates again strongly increases, whereas *DCP*'s performance is unaffected.

5.3. Caltech256

To show the possibilities that *DCP* offers for the exploration of image datasets, we implemented a light-weight interactive system that allows us to choose datasets and context classes from the Caltech256 dataset[3][10] and to visually inspect the resulting most discriminative splits. Internally, each image in Caltech256 is represented by a 1000-bin bag-of-visual words histogram based on quantized SIFT descriptors. In general, the classes in Caltech256 are very uniform and clustering algorithms are not able to identify meaningful subclusters. Here it comes as a benefit that *DCP* has access to a large number of canonical context classes, namely all other Caltech256 categories. While many of those yield very imbalanced splits, making *DCP* behave similar to an outlier detector, some combinations yield balanced and interpretable results. Figure 6 shows two of these: the *brain* class is split using two different contexts, *breadmaker* and *desk globe*. While internally the split are just based on *DCP*'s maximum separation property, a human user can give a semantic interpretation, *e.g.* a distinction between smooth and structures brain images in Figure 6(a), and a distinction between natural and artificial brain images in Figure 6(b).



(a) Split of `020.brain` with `021.breadmaker` context



(b) Split of `020.brain` with `053.desk-globe` context

Figure 6. Explorative use of *DCP*: the *brain* class in Caltech256 is split using two different context classes. A human could interpret the first split as *structured vs. smooth*, and the second split as *natural vs. schematic*.

Of course, this characterization is subjective, as the whole process of partitioning without labels is. *DCP* provides a mechanism to come up many different splits—one per context class—that give the user a meaningful selection of all partitioning, based on a well-defined maximization criterion. A typical clustering algorithm would require a change of the data representation or the pair-wise distance function for this, which is hardly possible in an interactive system that is meant to be usable by non-experts.

[3] `http://www.vision.caltech.edu/Image_Datasets/Caltech256/`

6. Summary and Outlook

We have proposed *discriminant context partitioning (DCP)*, a new approach to split a dataset into a fixed number of parts. It differs from *clustering* techniques as it can also identify parts that do not correspond to regions of high sample density in feature space. This is because it relies on a purely discriminative criterion, namely how well the parts can be separated from a disjoint dataset of "non-examples" using a large margin classifier. *DCP* is motivated by the observation that the human way of splitting a dataset into *interesting parts* is not necessarily guided by which patterns occur most frequently. Instead, it is natural for humans to characterize parts of a dataset by how easily they can be detected and described in a realistic environment with objects of other classes acting as reference. As a main contribution, we gave a large-margin formulation of this concept, and showed that the resulting optimization problem is very close to well-known approaches from semi-supervised machine learning. This allowed us to come up with two algorithms that rely on standard optimization techniques.

We showed promising results on images datasets of handwritten digits, and on a subset of the Caltech256 collection. Because *DCP* avoids the almost ubiquitous assumption that partitioning a set is the same as identifying peaks in the sample density, it is also applicable to unimodal and strongly imbalanced dataset, yielding much more stable results than classical methods like K-means.

DCP requires access to a context class, and different contexts give rise to different splits of the data. *DCP* is therefore not intended to replace existing clustering algorithms in situations, where the clustering objective is a well defined intrinsic property of the dataset, such as vector quantization. Instead, the approach of partitioning with a context class can serve as a tool to explore the track that clustering is a subjective process, which depends not only on the dataset itself but also on which aspect the user is interested in. Using *DCP*, one can generate different meaningful splits of the data by only changing the context set, but without the need to come up with a new data representation or to define a new distance function. This allows an explorative and interactive way of dealing with the partitioning problem, and in particular one that is also accessible to non-experts.

For future work, we see the need to improve the understanding of how data and context interact. This could allow to come up with new suitable classifiers and kernels. Also, we plan to further study how human make use of context and how this can be used to find better automatic algorithms.

Acknowledgments

This work was funded in part by the EC project CLASS, IST 027978.

References

[1] F. Aiolli and A. Sperduti. Multiclass Classification with Multi-Prototype SVMs. *JMLR*, 6, 2005.

[2] M. A. Aizerman, E. M. Braverman, and L. I. Rozonoér. Theoretical Foundations of the Potential Function Method in Pattern Recognition Learning. *Automation and Remote Control*, 25, 1964.

[3] A. Astorino and A. Fuduli. Nonsmooth Optimization Techniques for Semisupervised Classification. *PAMI*, 29(12):2135–2142, 2007.

[4] A. Ben-Hur, D. Horn, H. T. Siegelmann, and V. Vapnik. Support Vector Clustering. *JMLR*, 2:125–137, 2001.

[5] K. P. Bennett, P. S. Bradley, and A. Demiriz. Constrained K-Means Clustering. Technical Report 65, Microsoft, 2000.

[6] O. Chapelle. Training a Support Vector Machine in the Primal. *Neural Computation*, 19(5), 2007.

[7] O. Chapelle and A. Zien. Semi-Supervised Classification by Low Density Separation. In *AISTATS*, pages 57–64, 2005.

[8] Y. Cheng. Mean Shift, Mode Seeking, and Clustering. *PAMI*, 17(8):790–799, 1995.

[9] K. Fukunaga and L. D. Hostetler. The Estimation of the Gradient of a Density Function, with Applications in Pattern Recognition. *Information Theory*, 21(1):32–40, 1975.

[10] G. Griffin, A. Holub, and P. Perona. Caltech-256 Object Category Dataset. Technical Report 7694, California Institute of Technology, 2007.

[11] R. A. Jacobs, M. I. Jordan, S. J. Nowlan, and G. E. Hinton. Adaptive Mixtures of Local Experts. *Neural Computation*, 3(1), 1991.

[12] A. K. Jain and R. C. Dubes. *Algorithms for Clustering Data*. Prentice Hall, 1988.

[13] J. MacQueen. Some Methods for Classification and Analysis of Multivariate Observations. In *Proc. of the 5th Berkeley Symp. on Mathematics Statistics and Probability*, 1967.

[14] A. Y. Ng, M. Jordan, and Y. Weiss. On Spectral Clustering: Analysis and an algorithm. In *NIPS 14*, 2002.

[15] J. Puzicha, T. Hofmann, and J. Buhmann. A theory of proximity based clustering. *Pattern Recognition*, 33, 2000.

[16] R. M. Rifkin. *Everything Old is New Again : a Fresh Look at Historical Approaches in Machine Learning*. PhD thesis, Massachusetts Institute of Technology, 2002.

[17] B. Schölkopf, R. Herbrich, and A. J. Smola. A Generalized Representer Theorem. In *COLT*, 2001.

[18] Y. A. Sheikh, E.Khan, and T. Kanade. Mode-Seeking by Medoidshifts. In *ICCV*, 2007.

[19] J. Shi and J. Malik. Normalized Cuts and Image Segmentation. *PAMI*, 2000.

[20] V. Sindhwani, S. S. Keerthi, and O. Chapelle. Deterministic Annealing for Semi-Supervised Kernel Machines. In *ICML*, 2006.

[21] V. Vapnik. *Statistical Learning Theory*. Wiley, 1998.

[22] K. Wagstaff and C. Cardie. Clustering with Instance-Level Constraints. In *ICML*, 2000.

[23] J. Weston, R. Collobert, F. H. Sinz, L. Bottou, and V. Vapnik. Inference with the Universum. In *ICML*, 2006.

[24] L. Xu, J. Neufeld, B. Larson, and D. Schuurmans. Maximum Margin Clustering. In *NIPS 16*, 2004.