Data Min Knowl Disc (2010) 21:130–152
DOI 10.1007/s10618-010-0170-1

Learning in parallel universes

Bernd Wiswedel · Frank Höppner · Michael R. Berthold

Received: 21 March 2009 / Accepted: 5 March 2010 / Published online: 25 May 2010
The Author(s) 2010

Abstract We discuss *Learning in parallel universes* as a learning concept that encompasses the simultaneous analysis from multiple descriptor spaces. In contrast to existing approaches, this approach constructs a global model that is based on only partially applicable, local models in each descriptor space. We present some application scenarios and compare this learning strategy to other approaches on learning in multiple descriptor spaces. As a representative for learning in parallel universes we introduce different extensions to a family of unsupervised fuzzy clustering algorithms and evaluate their performance on an artificial data set and a benchmark of 3D objects.

Keywords Parallel universes · Descriptor space · Clustering

1 Introduction

Computer-driven learning techniques are based on a suitable data-representation of the objects being analyzed. The classical concepts and techniques that are typically applied are all based on the assumption of an appropriate single representation. This represen-

Responsible editor: Johannes Fürnkranz and Arno Knobbe.

B. Wiswedel (✉) · M. R. Berthold
Department of Computer and Information Science, University of Konstanz, Konstanz, Germany
e-mail: Bernd.Wiswedel@uni-konstanz.de

M. R. Berthold
e-mail: Michael.Berthold@uni-konstanz.de

F. Höppner
Faculty of Business (Information Systems), Ostfalia University of Applied Sciences, Wolfenbüttel, Germany
e-mail: f.hoeppner@ostfalia.de

tation, typically a vector of numeric or nominal attributes, is assumed to sufficiently describe the underlying objects. In many application domains, however, it is possible to derive multiple descriptions for an object. The resulting different descriptor spaces typically reflect different characteristics of the underlying objects and as such often even have their own unique semantics.

As most classical learning techniques are restricted to learning in exactly one descriptor space, learning in the presence of several object representations is in practice often solved by using one of the following three schemes.

1. Reduce the analysis to the analysis of any one of the descriptors. This is a rather strong limitation as it potentially ignores the information that is encoded in the remaining object representations.
2. Construct a joint descriptor space by, e.g. concatenating the features of the individual descriptor spaces. Such a combination is often impossible due to different feature domains. It may also introduce artifacts as it blurs the information that is present in different object representations.
3. Perform an analysis of each space, one by one, and fuse the individual models afterwards. This ad hoc solution delays the task of merging results to a post processing step. It has the major drawback of not paying attention to possible structural overlaps among descriptor spaces since the individual model construction is carried out independently. These overlaps, however, are often of special interest and, if detected during model construction, can be matched in order to reach consensus, for example.

All these strategies have limitations because they either ignore or obscure the multiple facets of the objects (given by the descriptor spaces) or do not respect overlaps. This often makes them inappropriate for practical problems.

We propose *learning in parallel universes* as a novel learning scheme that encompasses the simultaneous analysis of all given descriptor spaces, hereafter referred to as *universes*. The new learning scheme deals with the concurrent generation of local models in each universe, whereby these models cover local structures that are unique for individual universes and at the same time can also cover other structures that span multiple universes. This is in contrast to the goal of existing multi-view learning methods, which aim to build one global model from synergies between different universes (we will discuss the difference to state-of-the-art techniques in more detail in the next section). The main objective of learning in parallel universes is therefore the construction of a global model based on these local models that outperforms the above-mentioned schemes and provides new insights with regard to overlapping as well as universe-specific patterns. The learning task itself can be both supervised (e.g. building a classifier) and unsupervised (e.g. clustering). For supervised learning tasks, the model quality can be assessed using classical quality measures, for instance the classification accuracy on a test data set. One reasonable lower bound for this quality measure is the accuracy of a model that is trained using any of the schemes outlined above, i.e. based on the joint descriptor space or based on individual universes. Providing such a lower bound for unsupervised learning tasks is considerably more difficult since there is no target attribute. In these cases it requires an expert to evaluate the results or the availability of some reference clustering against which it can be compared, for instance, by using some entropy-based quality measures. However, the prime interest of cluster analysis is usually the identification of interesting groupings rather than the optimization of some numeric quality function. For these purposes the representation of clusters using soft membership values has proven to be useful since it enables a better interpretability over crisp clustering algorithms such as k-means (Klawonn 2004).

In this article we focus on unsupervised clustering in parallel universes and further develop previous work of Wiswedel and Berthold (2007). We present different extensions of the classical fuzzy c-means and possibilistic clustering algorithms (Bezdek 1981; Krishnapuram and Keller 1993) to allow for a simultaneous clustering in different universes. The key idea is to augment the characteristic object functions by additional terms that represent membership values of clusters to universes. Similar to the classical single-universe algorithms, these newly introduced membership values can be interpreted probabilistically (as in fuzzy c-means) or possibilistically, whereby the latter is less sensitive to an inadequate choice of the cluster count parameter and often allows for a more natural clustering as it does not have the partitioning property of the classical fuzzy c-means. The presented extensions specifically address some drawbacks of our previous work, which include a poor interpretability of the results and an unintuitive cluster count specification. The resulting clustering across different universes can be seen as a combination of local models (e.g. clusters) in one or several universes which, combined, form the global model.

The remainder of this article is organized as follows. In the next section we introduce learning in parallel universes and discuss some typical application scenarios. In the following section we present and distinguish the proposed method against related work. In Sect. 4 we review the classical fuzzy c-means algorithm and the possibilistic clustering approach as well as our previous work, which also focused on clustering in parallel universes. We derive new clustering schemes for learning in parallel universes in Sect. 5 and conclude with the presentation of results on artificial data and a 3D benchmark data set in Sect. 6.

2 Learning in parallel universes

Learning in parallel universes refers to the problem-setting of learning from objects given in multiple descriptor spaces, whereby single descriptors are potentially not sufficient for learning. More formally, we consider it as learning from a set of objects, each object being described in U, $1 \leq u \leq U$, parallel universes. The object representation for object i in universe u is a vector $\mathbf{x}_{i,u}$, often a (high-dimensional) vector of scalars. Note that the notion of a similarity or distance function between objects—which is a fundamental requirement for, e.g. clustering or distance-based classification tasks—can only be defined with regard to a specific universe and not the general object domain. The distance function between two objects i and j in universe u is $d_u\left(\mathbf{x}_{i,u}, \mathbf{x}_{j,u}\right)$. Throughout the paper we will use the term *object* to denote the objects being analyzed, whereby *object representation* or *instance* shall denote the (typically numerical) description of an object in a certain universe. The task of learning in parallel universes describes the creation of a global model that combines local models in individual universes, which may only explain part of the ground truth underlying the data. The identification of overlaps (or lack thereof) between different local models by a simultaneous analysis of all available universes is an important property of the learning process. The learning objective is twofold:

1. Identify structures that occur only in one or few universes (for instance groups of objects that cluster well in one universe but not in others).
2. Detect and facilitate overlapping structures between multiple, not necessarily all, universes (for instance clusters that group in multiple but not necessarily all universes).

Item (1) addresses the fact that any one universe may not be a complete representation of the underlying objects, that is, it does not suffice for learning. The task of descriptor generation is in many application domains a science by itself, whereby single descriptors (i.e. universes) carry a semantic and mirror only certain properties of the objects.

Item (2) describes cases, in which structures overlap across different universes. For unsupervised clustering tasks, for instance, this would translate to the identification of groups of objects that are self-similar in a subset of the universes. Note that in order to detect these overlaps and to support their formation it is necessary for information to be exchanged between the individual models during their construction. This is a major characteristic of learning in parallel universes.

Learning in parallel universes concerns the construction of local models for individual universes, whereby information is shared between the universes during model construction. This helps to create superior local models, which are eventually combined into a global model that covers all structures available in the universe. The very important point here is that individual local models do not necessarily cover all information in each individual universe.

2.1 Application scenarios

The challenge of learning in parallel universes can be found in almost all applications that deal with the analysis of complex objects. We outline a few of these below.

2.1.1 Molecular data

This includes activity prediction of drug candidates, typically small molecules. Molecules can be described in various ways, potentially focusing on different aspects such as surface charge distribution, shape, or flexibility (Bender and Glen 2004). It can be as simple as a vector of numerical properties such as molecular weight or number of rotatable bonds. Other descriptors concentrate on more specific properties such as charge distribution, 3D conformation or structural information. Apart from such semantic differences, descriptors can also be of a different type including vectors of scalars, chemical fingerprints (bit vectors) or graph representations. These diverse representations make it impossible to simply join the descriptors and construct a joint feature space.

It is known that molecular activity, e.g. the inhibition of a particular disease, is often caused by different factors. For instance, a molecule can bind to a certain protein and hence show activity because of its 3D layout (basically hooking into the binding pocket), whereas other molecules just have the right surface charge distribution. It is obvious that in such cases two different universes, one describing charge properties and another for the 3D layout, are necessary to model those two scenarios. At the same time it may also be of interest to identify groups that overlap in these universes as they may have a high potential of being a good drug candidate.

2.1.2 3D object data

Another interesting application is the mining of 3D objects (Bustos et al. 2006). The literature lists three main categories of descriptors: (1) image-based (features describing projections of the object, e.g. silhouettes and contours), (2) shape-based (like curvature measures) and (3) volume-based (partitioning the space into small volume elements and then considering elements that are completely occupied by the object or contain parts of its surface area, for instance). Which of these descriptions is appropriate for learning, mostly depends on the class of object being considered. For instance image- or volume-based descriptors fail on modeling a class of humans taking different poses, as their projections or volumes differ, whereas shape-based descriptors have proven to cover this class nicely. On the other hand classes such as cars and couches, which have no easily distinctive characteristic and distinguishable shape, are covered well in image- or volume-based descriptors.

2.1.3 Image data

Similarly, there are also manifold techniques available to describe images. Descriptors can be based on properties of the color or gray value distribution; they can encode texture information or edge properties. Other universes can reflect user annotations such as titles that are associated with the image. Also, in this domain it sometimes depends on the descriptor whether two images are similar or not. As an example consider a (green) billiard table with colorful billiard balls. This is likely to be similar to a meadow of flowers in a color-based universe, and yet they will be quite different in a universe that reflects edge information due to the different homogeneous areas.

3 Related work

A variety of work in related research fields also focuses on the analysis or construction of several feature spaces. The most prominent ones are described below.

3.1 Clustering high-dimensional data

Methods for clustering in high-dimensional data (Kriegel et al. 2009) operate on a single, typically very high-dimensional input space. There are a number of different subcategories, including methods for finding clusters in (mostly axis-parallel) subspaces and correlation clustering. The goal of so-called subspace clustering methods is to identify regions of the input space (a set of features) that exhibit a high similarity on a subset of the objects, or, more precisely, the objects' data representations. Subspace clustering methods are commonly categorized into bottom-up and top-down approaches. Bottom-up starts by considering density on individual features and then subsequently merging features/subspaces to subspaces of higher dimensionality, which still retain a high density. The most prominent example is CLIQUE (Agrawal et al. 1998), which partitions the input space using a static grid and then merges grid elements if they meet a given density constraint. Top-down techniques initially consider the entire feature space and then remove irrelevant features from clusters to compact the resulting subspace clusters. Friedman and Meulmany (2004) propose COSA, a general framework for this type of subspace clustering; they use weights to encode the importance of features to subspace clusters. These weights influence the distance calculation and are optimized as part of the clustering procedure. Another example for the top-down approach is PROCLUS (Aggarwal et al. 1999), which picks cluster prototypes from a set of candidates. The subspace assignment is accomplished using the features along which the objects in a cluster in the full feature space have the least spread. Another categorization of these algorithms is what Kriegel et al. (2009) call a *problem-oriented* view and the way the methods respect overlaps between clusters. They distinguish between subspace clustering (possible overlaps between subspace clusters, e.g. used in CLIQUE), projective clustering (unique assignment of objects to clusters + possibly noise, e.g. PROCLUS), and hybrid methods. The latter are somewhere in between, for instance COSA does not solve the clustering problem directly but defines a distance function for the application of classical distance-based clustering methods for the problem at hand.

Correlation clustering methods extend the concept of subspace clustering by finding clusters that form on arbitrarily oriented subspaces, often hyperplanes constructed from the local application of principal component analysis (PCA). ORCLUS (Aggarwal and Yu 2000) extends the idea of PROCLUS by performing a PCA on prototype clusters on the full feature space and then choosing the weakest eigenvectors to span the assigned subspace as these eigenvectors reveal a high density.

What these types of methods for clustering in high-dimensional spaces share with learning in parallel universes is that they also try to respect locality in terms of the features space—although these methods follow a very different goal. They refer to individual features or feature spaces derived from a purely numeric (usually correlation) analysis whereas we consider universes, i.e. semantically meaningful descriptor spaces.

3.2 Multi-view-learning

In multi-view-learning (Rüping and Scheffer 2005), a similar setup to learning in parallel universes is assumed, i.e. the availability of multiple descriptor spaces (universes or *views*). Most multi-view learning methods have a very focused learning scope and expect all universes/views to share the same structure as they promote a consensus between views. Therefore each individual universe would suffice for learning if enough or distortion free training data were available. Some of the first work in the multi-view area was done by Blum and Mitchell (1998), who concentrate on the classification of web pages. They introduce *co-training* in a semi-supervised setting, i.e. they have a relatively small set of labeled and a rather large set of unlabeled instances. The web pages are described using two different views, one based on the actual content of the web site (bag of words), the other one based on anchor texts of hyperlinks pointing to that site—both views are assumed to be *conditionally independent*. Co-training creates a model on each view, whereby the training set is augmented with data that has been labeled with high confidence by the model of the respective other view. This way the two classifiers bootstrap each other. Dasgupta et al. (2001) provide the theoretical foundation for the co-training setup. They show that the disagreement rate between two independent views is an upper bound for the error rate of either hypothesis. This principle already highlights the identical structure property of all views as the base assumption of many multi-view learners, which is contrary to the setup of learning in parallel universes, where some information may only be available in a subset of universes.

Unsupervised clustering methods for multi-view problems have been discussed by Kailing et al. (2004) and Bickel and Scheffer (2004). Kailing et al. extend the DBSCAN algorithm to work on multiple views. The classical DBSCAN algorithm uses the notion of dense regions by means of core objects, i.e. objects that have a minimum number k of objects in their (ϵ-) neighborhood. A cluster is then defined as a set of (connected) dense regions. The multi-view version of DBSCAN extends the dense region definition by requiring that an object is a neighbor to a core object if it is in its ϵ-neighborhood in (1) at least one or (2) all views, whereby only case (2) can be considered a classical multi-view learning setup as it assumes consensus between the views. Bickel and Scheffer (2004) present extensions to the EM algorithm, the k-means and the agglomerative clustering algorithms for multi-view problems. They concentrate on the clustering of text data using two views. The main idea of their extensions is to alternately exchange intermediate results between the views, e.g. the multi-view version to k-means iteratively uses the partition information from one view to determine the next set of mean vectors in the other view. Alqadah and Bhatnagar (2008) introduce the term *3-clusters* to denote subspace clusters, which overlap in different views. They mine binary data and seek to identify clusters, which are maximal in both the number of covered objects and the number of attributes describing the cluster in different views.

Classical multi-view methods assume all views to have the same structural information, i.e. they promote a consensus between different universes. In spite of this interesting learning challenge, it turns out that in many application domains the different universes often reflect—by design—different aspects of the underlying objects. In these cases the information that is (hiddenly) encoded in individual universes often only partially overlaps, thus violating the prime assumption of multi-view-learning and making these methods inapplicable.

Learning in parallel universes might be viewed as a variant of multi-view learning, because both approaches start with multiple views on the data. The main difference is that the multi-view models currently developed in the literature, all induce a single global model that is global in all views/universes, whereas our approach combines local models from individual views/universes into a model that is global across all universes. The resulting model is therefore not applicable in each individual universe, but only across all universes. In order to emphasize this difference, we prefer to call it *Learning in parallel universes*, a term which was first introduced by Patterson and Berthold (2001).

There exist a number of other similar learning setups. These include ensemble learning, multi-instance-learning and sensor fusion. We do not discuss these methods in detail but point out that they all have either a different learning input (no a-priori defined universes) or a distinct learning focus (no partially overlapping structures across universes while retaining universe semantics).

4 Clustering in parallel universes

In the following we present new clustering schemes based on the classical fuzzy c-means (FCM) and possibilistic clustering (PCM) algorithms (Bezdek 1981; Krishnapuram and Keller 1993). We quickly review both algorithms as we use concepts from them later on, and we start by introducing the necessary notation and reviewing the techniques underlying the classical one-universe algorithms. We summarize earlier work (Wiswedel and Berthold 2007) and, in the next section, present the main contribution of this article, i.e. a new clustering algorithm for parallel universes that addresses some drawbacks of our previous work.

As stated previously, we examine a set of objects, each object being described in U parallel universes. We denote the overall number of objects as P. The object representation for object i, $1 \leq i \leq P$, in universe u, $1 \leq u \leq U$, is $\mathbf{x}_{i,u}$. We are interested in identifying K cluster. We further assume appropriate definitions of distance functions for each universe $d_u\left(\mathbf{w}_{k,u}, \mathbf{x}_{i,u}\right)$, whereby $\mathbf{w}_{k,u}$ denotes the k-th prototype in the u-th universe.

4.1 Classical FCM & PCM in one universe

We briefly review the standard fuzzy c-means and possibilistic clustering in a single universe, i.e. one feature space only, since we will use ideas from both techniques in the following sections. For the sake of clarity the subscript u is omitted in the following. We also do not give the update equations that minimize the respective target functions but instead refer to the literature. Both FCM and PCM iteratively minimize an objective function that represents the weighted sum of intra-cluster distances. The lower the sum, the better the clustering. The FCM objective function is (Bezdek 1981):

$$\min_{v_{i,k},\mathbf{w}_k} \sum_{i=1}^{P}\sum_{k=1}^{K} v_{i,k}^{m}\, d\left(\mathbf{w}_k, \mathbf{x}_i\right) \qquad \text{s.t.} \quad \forall i : \sum_{k=1}^{K} v_{i,k} = 1 \,. \tag{1}$$

The objective function represents the accumulated sum of distances between object representations (instances) $\mathbf{x}_i$ and cluster centers $\mathbf{w}_k$, weighted by the degree of membership to which an object belongs to a cluster. The coefficient $m \in (1, \infty)$ is a fuzzification parameter and $v_{i,k}$ the respective value from the partition matrix, i. e. the degree to which an object representation $\mathbf{x}_i$ belongs to cluster k. The objective function is alternately optimized with respect to $v_{i,k}$ and $\mathbf{w}_k$, whereby the side constraint requires the objects to be shared among the clusters. This partitioning property is often a serious drawback of FCM as outliers and noise can heavily influence the clustering result.

In response to this problem, Krishnapuram and Keller (1993) presented possibilistic clustering (PCM), which does not have such a side constraint. The PCM objective function is (Krishnapuram and Keller 1993):

$$\min_{v_{i,k},\mathbf{w}_k} \sum_{k=1}^{K} \sum_{i=1}^{P} v_{i,k}^m \, d\left(\mathbf{w}_k, \mathbf{x}_i\right) + \sum_{k=1}^{K} \eta_k \sum_{i=1}^{P} \left(1 - v_{i,k}\right)^m \tag{2}$$

PCM also accumulates weighted distances between instances $\mathbf{x}_i$ and prototypes $\mathbf{w}_k$, however instead of distributing an object among the clusters it uses a penalty term η_k to avoid the trivial solution in which all clusters are empty. For each membership $v_{i,k}$ of an object i to a cluster k it uses a non-membership value $1 - v_{i,k}$, which will take an accordingly large value if an object has a large distance to the cluster prototype. This non-membership value is used to weight the penalty term η_k. Equation (2) is optimized w.r.t. $\mathbf{w}_k$ and $v_{i,k}$; the penalty term η_k is a parameter of the algorithm, which is determined using certain heuristics. Krishnapuram and Keller (1993) suggest using the average intra-cluster distance of cluster k, e.g. by calculating the average distance of instances with a high membership to the cluster prototype.

Despite the advantage of being less sensitive to noise, PCM has, in its single universe usage, a limitation, which becomes obvious when reformulating the objective function:

$$\min_{v_{i,k},\mathbf{w}_k} \sum_{k=1}^{K} \underbrace{\sum_{i=1}^{P} v_{i,k}^m \, d\left(\mathbf{w}_k, \mathbf{x}_i\right) + \eta_k \sum_{i=1}^{P} \left(1 - v_{i,k}\right)^m}_{\text{Optimization of cluster}\, k} \tag{3}$$

Obviously it can also be phrased as an independent optimization of each cluster. In fact, the objective function takes its theoretical minimum when all clusters represent the same minimal cluster and are therefore identical. This problem also often occurs in practice which is why Krishnapuram and Keller (1996) suggest different tricks, e.g. running a standard FCM to initialize the clusters and to choose a smaller fuzzifier m than is normally used in FCM.

Interestingly, as we will see later, this problem disappears when applied in parallel universes.

4.2 Fuzzy clustering in parallel universes

Both FCM and PCM operate on a single feature space and are therefore not directly applicable for problems described in parallel universes. In our previous work

(Wiswedel and Berthold 2007) we suggested modifying the objective function and including additional terms to represent the universes. This is accomplished by using new membership values $0 \le z_{i,u} \le 1$, which denote object memberships to universes and which are also optimized during the training process. If this type of object membership is learned as being small, i.e. close to 0, the object i is said to not contribute to any of the clusters in universe u, whereas a larger value indicates a substantial contribution to one of the clusters in u. The objective function is in the form of:

$$\min_{v_{i,k}^{(u)},\, z_{i,u},\, \mathbf{w}_{k,u}} \quad \sum_{i=1}^{P} \sum_{u=1}^{U} z_{i,u}^{m'} \sum_{k=1}^{K_u} \left(v_{i,k}^{(u)} \right)^m d_u \left(\mathbf{x}_{i,u}, \mathbf{w}_{k,u} \right)$$
$$\text{s.t.} \quad \forall i,\ u : \sum_{k=1}^{K_u} v_{i,k}^{(u)} = 1 \quad \forall i : \sum_{u=1}^{U} z_{i,u} = 1. \tag{4}$$

The input parameters for the objective function are appropriate definitions of the universes and their distance functions, the fuzzification parameters m and m' and the cluster counts K_u. The objective function can also been seen as a connected optimization of a fuzzy c-means in the individual universes (the inner term in Eq. 4), whereby the impact of object i to the clustering process in universe u is scaled by its membership value $z_{i,u}$. The objective function is optimized with respect to the cluster prototypes $\mathbf{w}_{k,u}$ and the membership values of objects to clusters $v_{i,k}^{(u)}$ and objects to universes $z_{i,u}$. The update functions that lead to a local minima of the objective functions are listed below (assuming Euclidean distance functions d_u for the update function (5c)):

$$v_{i,k}^{(u)} = \left(\sum_{\bar{k}=1}^{K_u} \left(\frac{d_u \left(\mathbf{x}_{i,u}, \mathbf{w}_{k,u} \right)}{d_u \left(\mathbf{x}_{i,u}, \mathbf{w}_{\bar{k},u} \right)} \right)^{\frac{1}{m-1}} \right)^{-1} \tag{5a}$$

$$z_{i,u} = \left(\sum_{\bar{u}=1}^{U} \left(\frac{\sum_{k=1}^{K_u} \left(v_{i,k}^{(u)} \right)^m d_u \left(\mathbf{x}_{i,u}, \mathbf{w}_{k,u} \right)}{\sum_{k=1}^{K_{\bar{u}}} \left(v_{i,k}^{(\bar{u})} \right)^m d_{\bar{u}} \left(\mathbf{x}_{i,\bar{u}}, \mathbf{w}_{k,\bar{u}} \right)} \right)^{\frac{1}{m'-1}} \right)^{-1} \tag{5b}$$

$$\mathbf{w}_{k,u} = \frac{\sum_{i=1}^{P} z_{i,u}^{m'} \left(v_{i,k}^{(u)} \right)^m \mathbf{x}_{i,u}}{\sum_{i=1}^{P} z_{i,u}^{m'} \left(v_{i,k}^{(u)} \right)^m}. \tag{5c}$$

The algorithm that uses these update functions is shown in Algorithm 1. It starts with a random initialization of the prototypes and then iteratively optimizes the variables according to the update functions to learn a (local) minima.

Although this extension already enables a simultaneous analysis of different universes, it has a few drawbacks:

– The universes compete for the objects since the membership degrees of objects to universes need to sum to one (second side condition in Eq. 4). This is specifically

Algorithm 1 Fuzzy c-means algorithm for parallel universes (Wiswedel and Berthold 2007)

```
   /*input parameters:*/
   /*$K_u$: cluster count per universe*/
   /*$m, m'$: fuzzifiers $m, m' > 1$*/
1: $\forall u, k \in [1, K_u]$: initialize prototypes $\mathbf{w}_{k,u}$ by assigning them to random data points $\mathbf{x}_{i,u}$
2: $\forall i, u$: initialize the membership values $z_{i,u} = \frac{1}{U}$
3: repeat
4:    $\forall i, u, k \in [1, K_u]$: compute $v_{i,k}^{(u)}$ according to (5a)
5:    $\forall i, u$: compute $z_{i,u}$ according to (5b)
6:    $\forall u, k \in [1, K_u]$: compute prototypes $\mathbf{w}_{k,u}$ according to (5c)
7: until termination, e.g. when no significant change to partitioning matrices $v_{i,k}^{(u)}$ or $z_{i,u}$
8: return partitioning values $v_{i,k}^{(u)}$, $z_{i,u}$ and cluster prototypes $\mathbf{w}_{k,u}$, $\forall i, u, k \in [1, K_u]$
```

problematic and counterintuitive for clusters that heavily overlap in different universes. Instead the objective function favors cluster that only arise in one universe.

– The user has to specify the number of clusters for each universe separately.
– Clusters in different universes are independent of each other. They are not interconnected and therefore it would also not be possible to identify universe overlapping information, i.e. objects that group well in multiple universes.
– The results are difficult to interpret because each cluster consist of its prototype ($\mathbf{w}_{k,u}$) and a set of objects, whereby these are accompanied by the cluster ($v_{i,k}^{(u)}$) and the universe membership ($z_{i,u}$).
– The clustering is affected by noise and outliers due to its partitioning property. Although the influence of noisy objects in a universe can be reduced when these objects cluster well in the remaining universes, they can still prove problematical if they do not contribute to any of the clusters.

These disadvantages limit the general usage of this approach since often there is no prior knowledge on how many clusters to expect in a given universe and also the missing connection between the universes is often inappropriate. These concerns can be addressed using a different interpretation of the membership values and a different formulation of the target function.

5 Clustering across parallel universes

In the previous section we used membership values $z_{i,u}$ that represent objects' memberships to universes, which will be high if the corresponding object representation is well covered by one of the clusters in a universe. However, these membership values only create a weak connection between different universes. In the following we use a different interpretation of these values, whereby we assign clusters, instead of objects, to universes. The objective functions discussed below identify clusters that are shared among different universe, i.e. each cluster has a representation in each universe. If a cluster is not well represented in a universe (the objects contained in the clusters exhibit a small self-similarity in that universe), the algorithm learns its corresponding membership $z_{i,u}$ is small. If the cluster is well represented in a universe, i.e.

the contained objects are very similar to each other, the membership of the cluster to the universe is accordingly large. This is a fundamental difference to the formulation discussed in the previous section since the clusters are the same across all universes but simply assigned to a different degree depending on how well the cluster forms in a universe. This approach is more natural since it allows a better interpretability of the results and an easier specification of the input parameters as there is only one global cluster count instead of counting the number of clusters in each universe. The learning algorithm can then adjust the cluster-to-universe membership to move clusters to universes where they are needed. We will discuss both a fuzzy and a possibilistic interpretation of these membership values.

5.1 Fuzzy clustering across parallel universes

In order to establish a direct connection between clusters in different universes we use the already introduced membership values to indicate a cluster membership to universes. That is, there is one fixed number of global clusters, whereby their significance to the universes is learned throughout the training. The new fuzzy clustering objective function has the following form:

$$\min_{v_{i,k},\, z_{k,u},\, \mathbf{w}_{k,u}} \quad \sum_{u=1}^{U} \sum_{k=1}^{K} z_{k,u}^{m'} \sum_{i=1}^{P} v_{i,k}^{m} d_u\left(\mathbf{x}_{i,u}, \mathbf{w}_{k,u}\right)$$

$$\text{s.t.} \quad \forall i,\, u : \sum_{k=1}^{K} v_{i,k} = 1 \quad \forall k : \sum_{u=1}^{U} z_{k,u} = 1. \tag{6}$$

The objective function represents a classical fuzzy c-means within each universe (inner sum of Eq. 6), whereby this sum of weighted distances is multiplied by the membership value $z_{k,u}$, i.e. the degree to which a cluster is represented in a universe. This scaling is analogous to the weighting of the distances in a classical c-means approach: $z_{k,u}$ (or $v_{i,k}$) will be large if the weighted sum of distances $\sum_{i=1}^{P} v_{i,k}^{m} d_u\left(\mathbf{x}_{i,u}, \mathbf{w}_{k,u}\right)$ (or distance $d_u\left(\mathbf{x}_{i,u}, \mathbf{w}_{k,u}\right)$) is small. The algorithm is no longer based on universe specific cluster counts, which also enables a better interpretability of the variables $z_{k,u}$. These are learned to be large, i.e. $z_{k,u} \to 1$ if a cluster k forms primarily in universe u. If these values are similar between different universes (i.e. similarly large), it indicates that this cluster forms in different universes equally well. It is important to note that the partitioning of objects to clusters, represented by $v_{i,k}$, is also carried out across all universes, whereas the objective function (4) does this for all universes individually. The output of this approach is a set of cluster prototypes along with their respective membership values and the objects with their membership values to the clusters. The update functions for the membership values and the cluster prototypes are:

$$v_{i,k} = \left(\sum_{\bar{k}=1}^{K} \left(\frac{\sum_{u=1}^{U} z_{k,u}^{m'} d_u\left(\mathbf{x}_{i,u}, \mathbf{w}_{k,u}\right)}{\sum_{u=1}^{U} z_{\bar{k},u}^{m'} d_u\left(\mathbf{x}_{i,u}, \mathbf{w}_{\bar{k},u}\right)} \right)^{\frac{1}{m-1}} \right)^{-1} \tag{7a}$$

$$z_{k,u} = \left(\sum_{\bar{u}=1}^{U} \left(\frac{\sum_{i=1}^{P} v_{i,k}^{m} d_u \left(\mathbf{x}_{i,u}, \mathbf{w}_{k,u} \right)}{\sum_{i=1}^{P} v_{i,k}^{m} d_{\bar{u}} \left(\mathbf{x}_{i,\bar{u}}, \mathbf{w}_{k,\bar{u}} \right)} \right)^{\frac{1}{m'-1}} \right)^{-1} \tag{7b}$$

$$\mathbf{w}_{k,u} = \frac{\sum_{i=1}^{P} v_{i,k}^{m} \, \mathbf{x}_{i,u}}{\sum_{i=1}^{P} v_{i,k}^{m}} \, . \tag{7c}$$

The objective function (6) already has certain advantages over the approach discussed in the previous section as it establishes a direct connection between different universes. However, due to its partitioning property it can still be heavily affected by noisy objects and outliers. Additionally, the second side constraint requires the cluster-to-universe memberships to sum to 1, which is particularly unsuited if clusters heavily overlap across universes. In such cases it would be more appropriate to allow high membership values ($z_{k,u} \to 1$) in multiple universes simultaneously.

5.2 Possibilistic clustering across parallel universes

These requirements can be met using a possibilistic interpretation of the membership values $z_{k,u}$, i.e. by dropping the second side constraint in Eq. 6. The new objective function is given in Eq. 8; it uses the objective function of standard fuzzy c-means within the individual universes, whereby they are linked using a possibilistic membership value to represent a cluster's typicality for a universe.

$$\min_{v_{i,k},\, z_{k,u},\, \mathbf{w}_{k,u}} \quad \sum_{u=1}^{U} \sum_{k=1}^{K} z_{k,u}^{m'} \sum_{i=1}^{P} v_{i,k}^{m} d_u \left(\mathbf{x}_{i,u}, \mathbf{w}_{k,u} \right) + \sum_{u=1}^{U} \eta_u \sum_{k=1}^{K} \left(1 - z_{k,u} \right)^{m'} \sum_{i=1}^{P} v_{i,k}^{m}$$

$$\text{s.t.} \quad \forall i : \sum_{k=1}^{K} v_{i,k} = 1. \tag{8}$$

The second term of the objective function corresponds to the possibilistic interpretation as in standard PCM (Eq. 2). It represents the non-membership, $\left(1 - z_{k,u}\right)$, of a cluster k to a universe u and serves to control the assignment of $z_{k,u}$, i.e. to avoid the trivial solution, in which all $z_{k,u}$ would be set to 0. The role of the multiplier $\sum_{i=1}^{P} v_{i,k}^{m}$ is important here as it penalizes clusters that do not belong to any universe. Ignoring this multiplier would otherwise attract all objects into these no-universe clusters. Similar to the classical PCM the second term is scaled by a universe specific parameter η_u. The value of this parameter can be heuristically determined using, for instance the average intra-cluster distance in a universe. That is

$$\eta_u = \kappa \, \frac{\sum_{i=1}^{P} \sum_{k=1}^{K} z_{k,u}^{m'} v_{i,k}^{m} \, d_u \left(\mathbf{x}_{i,u}, \mathbf{w}_{k,u} \right)}{\sum_{i=1}^{P} \sum_{k=1}^{K} z_{k,u}^{m'} v_{i,k}^{m}} \, , \tag{9}$$

whereby κ is a user parameter to scale this term, which is usually set to $\kappa = 2$. The update functions that lead to a minima of objective function (8) are given below (the update function for $\mathbf{w}_{k,u}$ is equal to (7c)):

$$z_{k,u} = \left(1 + \left(\frac{\sum_{i=1}^{P} v_{i,k}^{m} d_u \left(\mathbf{x}_{i,u}, \mathbf{w}_{k,u} \right)}{\eta_u \sum_{i=1}^{P} v_{i,k}^{m}} \right)^{\left(\frac{1}{m'-1}\right)} \right)^{-1} \tag{10a}$$

$$v_{i,k} = \left(\sum_{\bar{k}=1}^{K} \left(\frac{\sum_{u=1}^{U} z_{k,u}^{m'} d_u \left(\mathbf{x}_{i,u}, \mathbf{w}_{k,u} \right) + \left(1 - z_{k,u}\right)^{m'} \eta_u}{\sum_{u=1}^{U} z_{\bar{k},u}^{m'} d_u \left(\mathbf{x}_{i,u}, \mathbf{w}_{\bar{k},u} \right) + \left(1 - z_{\bar{k},u}\right)^{m'} \eta_u} \right)^{\frac{1}{m-1}} \right)^{-1}, \tag{10b}$$

In Sect. 4.1 we discussed the problem of identical clusters in the classical PCM. However, this problem is not present in the case of objective function (8), which also uses the possibilistic interpretation for $z_{k,u}$. Although Eq. 8 can be rewritten to an apparently independent optimization in each universe (similar to rephrasing the classical PCM target function (2) to Eq. 3) this does not mean that the optimization can be carried out independently. The key point here is the side constraint (the classical PCM in one universe does not have side constraints), which establishes a dependency between universes. The complete (unbounded) formulation of the optimization problem using a Lagrange function incorporates this side condition and can not be reformulated as independent optimization of the universes. This dependency becomes more obvious when looking at the final update Eq. 10a and b. The update function $z_{k,u}$ depends directly on the values of $v_{i,k}$, which themselves are based on *all* universes. More intuitively, in comparison to the single universes PCM, we cannot simply move cluster prototypes around to achieve a theoretical minimum of the objective function.

All methods discussed here are based on a suitable definition of the target function, which is minimized with respect to variables such as partitioning values and cluster prototypes. Even if the natures of all of these objective functions are similar, there is an important difference in the interpretation of the membership values z between the methodology discussed in our previous work (Wiswedel and Berthold 2007) and the objective functions discussed here. The new formulation assigns clusters to universes, which is more natural as it creates a direct link between different universes and allows clusters to be shared. Additionally, it simplifies the parameterization of the algorithm (there is one global cluster count) and makes the results more interpretable.

In the next section we will discuss the applicability of the presented methods on both an artificial data set and a real-world data set of 3D objects.

6 Results

We use two different data sets to demonstrate the usefulness of the presented approach. The first one is an artificial data set that was also used in our previous work. The second data set contains descriptions of 3D objects given in different universes, which cover volume-, image- and shape-based properties of the objects.

6.1 Artificial data

We first present some results on synthetic data with a variable number of universes in order to demonstrate the advantages and disadvantages of the presented approach and to compare it with our previous work. We use the same data generation process as described in (Wiswedel and Berthold 2007). The universes are two-dimensional feature spaces in order to allow a display of both data and generated clusters. In each universe we generate two Gaussian distributed clusters by using a total of 1,400 objects. We first randomly assign each of the objects to one of the universes, in which we draw the object representation (i.e. the object's features in the universe) according to the distribution of one of two clusters (randomly choosing one of the two). As each object is represented in each universe, we also generate features in the remaining universes using a uniform distribution, i.e. the object representations represent noise in these universes unless they are drawn by chance into an existing cluster.

Figure 1 shows an example data set with three universes. The top figures show the three universes, filtering only the object representations that were chosen to group in the respective universes. The overall six clusters are disjoint, two clusters belong to a universe, respectively. The clusters in the top figures are the reference clustering, i.e. the one that we try to recover in the following. The bottom figures show all objects, including the objects that cluster in the respective other universes. If we concentrate on a single universe only, it is hard to identify the two original clusters as $2/3$ of the data are noise in each universe.

This data generation process does not create any overlapping clusters across universes; each cluster forms solely in one universe. Due to its aforementioned characteristics the fuzzy clustering in parallel universe discussed in (Wiswedel and Berthold 2007) is very well suited to identify these hidden structures. Obviously the question

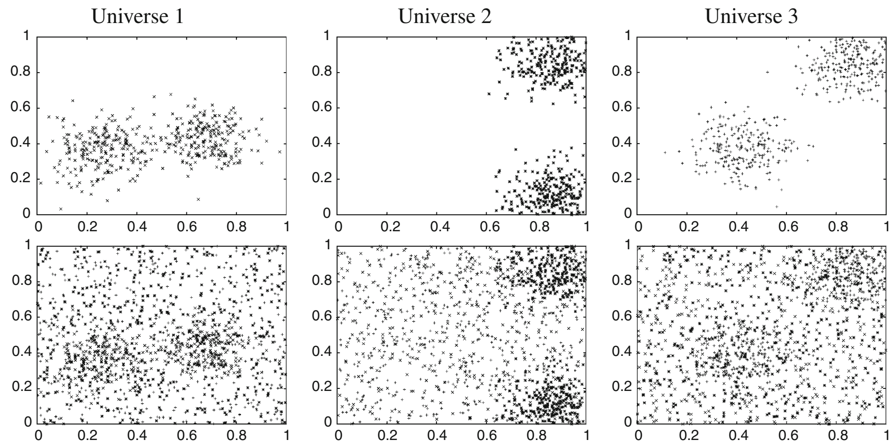


Fig. 1 Three universes of a synthetic data set. The *top* figures only show the object representations that were generated to cluster in the respective universe. The *bottom* figure shows all instances; about $^2/_3$ are noise in a universe. The objects originating from different clusters are shown in different shapes in order to make them distinguishable

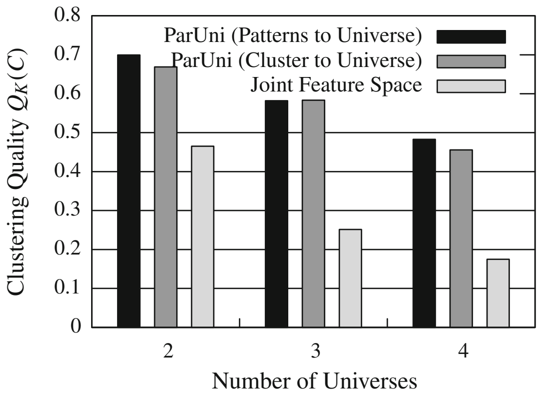


Fig. 2 Quality of clustering result on three data sets. The number of universes varies from two to four. The two bars on the *left* represent the fuzzy clustering using object and cluster memberships to universes, respectively. The bar on the *right* shows the clustering quality using a FCM on the joint features space. A decreasing quality with a increasing universe count is to be expected since the number of clusters also increases

arises whether the clustering methods that use membership values to assign clusters to universes are able to detect these clusters too.

Figure 2 compares the clustering results of a fuzzy clustering using object to universe memberships, a fuzzy clustering across universes assigning clusters to universes, and a standard FCM on the joint feature space. In each case the number of desired clusters was set appropriately, i.e. for the 3-universe problem it was set to identify two clusters per universe in the object to universe setup and overall six clusters for the fuzzy clustering using cluster memberships and the standard FCM.

The quality measure shown in the graph is calculated using an entropy based comparison of the computed clustering with the referencing clustering (Wiswedel and Berthold 2007):

$$Q_K(C) = \sum_{C_i \in C} \frac{|C_i|}{P} \cdot \big(1 - \text{entropy}_K(C_i)\big) \ ,$$

whereby K and C are the reference and computed clustering result and $\text{entropy}_K(C_i)$ denotes the entropy of cluster C_i with respect to K. The function evaluates to 1 if K and C are equal and 0 if C is random. The higher $Q_K(C)$ the better the clustering.

Both algorithms that exploit the universe information perform significantly better than the FCM on the joint feature space. Our new method using cluster memberships to universes achieves similar results to the clustering algorithm that assigns objects to universes. However, it is easier to set up due to its global cluster count parameter and it allows for interpretable results. When looking at the final membership values $z_{k,u}$ it becomes very obvious that each universe has two clusters, i.e. all clusters have a very high membership to only one of the universes and very low memberships to the remaining universes. This information is much harder to read from the output of the clustering algorithm that assigns objects to universes as here we would have to analyze all object memberships to see whether there is some intrinsic structure underlying them.

6.2 3D object data

In a second experiment we analyze a 3D object data base (3D Benchmark 2008; Bustos et al. 2004). There are a total of 292 objects, which were manually classified into 17 different groups such as airplanes, humans, swords, etc. The objects are described by means of four different universes, which can be categorized into image-, shape- and volume based descriptions. These are (Bustos et al. 2005):

DBF	is an image-based descriptor, which analyzes the six different projects of an object after normalizing and rotating it into a unit cube. This 366 dimensional universe respects distances between the object and the projection surface.
SIL	is also an image-based descriptor and is similar to DBF, although it uses only three projection planes instead of six and is based on an object's silhouettes. It has 510 attributes.
SSD	is a shape-based descriptor in 432 dimensions, taking into a account the curvature properties of the object surface.
VOX	is volume-based and reflects the properties of the object volume using voxels (small volume elements). It uses 343 dimensions to describe the distribution of voxels containing different fractions of the object surface.

We use the Euclidean norm to calculate distances between object representations and prototypes since the fuzzy clustering techniques require a differentiable distance function. As one part of the preprocessing we normalize the domain of the distance functions in order to have comparable distance ranges in all universes; we do so by normalizing the computed distances with the maximum distance in a universe. This is similar to the normalization applied in (Bustos et al. 2004). The final input parameters for the algorithm are therefore the overall cluster count as well as the fuzzifiers m, which controls the overlap of clusters, and m', controlling the formation of membership values $z_{k,u}$.

We investigated different settings for the cluster count, varying the value from 20 to 80. The quality values $Q_K(C)$ indicate that larger cluster counts yield numerically better clustering results, which is due to the nature of $Q_K(C)$ as it favors more clusters. The value of $Q_K(C)$ was sometimes above 0.95, whereby the large majority of clusters covered either singletons or groups of two or three objects of the same class, i.e. their entropy was 0. Only a few clusters ($\approx$10) covered entire classes, which apparently separate clearly from the remaining classes. These clusters were reproducibly detected, relatively independent on how the cluster count was chosen.

For the remaining experiments we set the overall cluster count parameter K to 30, as a result from the discussion above. This gives the learning algorithm 50% more clusters than there are classes in the dataset allowing for some tolerance against outliers in cluster/class assignments. In the following experiments we concentrate on the two remaining parameters, the fuzzifiers m and m', both of which need to be strictly larger than 1. We report the average quality retrieved from 10 runs in order to compensate for the random initialization and the resulting indeterministic clustering outcome. The fuzzifier m controls the overlap of clusters and corresponds directly to the fuzzifier used in the classical c-means setup, which offers a direct comparison to the clustering results in a single universe approach, i.e. the independent application of a classical c-means on the individual universes. Note also that the parallel universe approach assigning objects to universes (Wiswedel and Berthold 2007) is quickly rendered inappropriate since it requires a parameterization per universe and therefore some prior knowledge on the data set. Furthermore it lacks the interpretability of the results (cf. Sect. 4.2). Figure 3 compares the clustering qualities for different values of the fuzzifier m, whereby we show results for the Fuzzy Clustering across parallel universes and the Fuzzy c-means algorithm in the universes DBF, SIL, and SSD. For the parallel universe approach we chose the standard value $m' = 2.0$, which controls the cluster-to-universe membership assignment. There are different interesting observations in the graph in Fig. 3:

1. Even the single universe cluster qualities differ depending on which universe is used. This also corresponds to the results reported by Bustos et al. (2004), e.g. the DBF universe is the best performing universe for a majority of classes, whereas SSD performs generally badly except for selected classes (see also below).
2. The overall performance declines when the fuzzifier m is increased. The literature often suggests $m = 2$ as a good default value (Krishnapuram and Keller 1993; Bezdek 1981), though in the present case a much smaller value yields better results. This is certainly a characteristic of this data set and might be caused by the rather high dimensionality of each of the universes; a lower value m forces a crisp clustering and therefore reduces the impact of objects far away from a cluster center and hence the negative influence of the *curse of dimensionality* (Kriegel et al. 2009).
3. The parallel universe approach outperforms the single universe cluster method for all universes when choosing a small fuzzifier m but then drops even below the DBF and SIL line for larger values. As we will see later, this data set contains classes

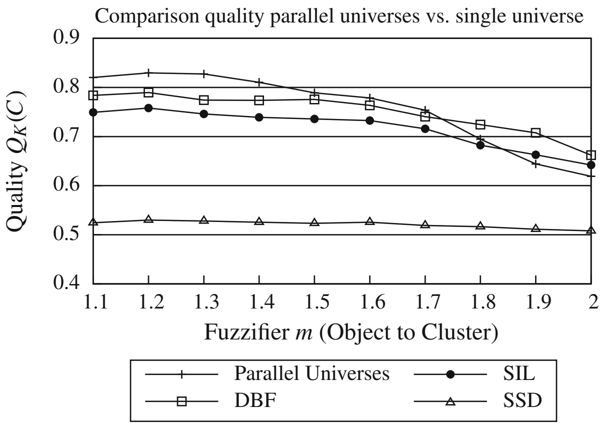


Fig. 3 Quality scores for parallel universe approach compared with results on three selected single universes

that group well in one or two universes only. In the case of large values m, however, we enforce more balanced membership values. These yield more uniform clusters, even though there might be no consensus between the universes for the cluster at hand. Small values of m result in more clearly separated and non-overlapping clusters (within a universe). Consequently, it is easier to detect whether clusters also occur in other universes and to assign adequate cluster-to-universe memberships $z_{k,u}$ in these cases only. This is a rather interesting observation, which we also noticed in our experiments: The formation of $z_{k,u}$ is not only (directly) influenced by the fuzzifier m', but also (indirectly) by how m is chosen.

In another set of experiments we analyzed the effect of different values for m' on the clustering result. This parameter controls the assignment of clusters to universes and is only defined for the parallel universe approach, hence we cannot draw any connections or make meaningful comparisons with the single universe algorithm. Larger values for m' produce more overlap between universes, whereas small values allow for more crisp assignments. Figure 4 summarizes the results for different values of m (represented by different curves). As already seen in the previous experiments, smaller values for m tend to give better clustering results, relatively independent on how m' is chosen. As can be seen in the figures, m' has, in comparison to m, far less impact. When choosing $m' \geq 2$ the curves for $m = 1.1$ and $m = 1.3$ (the ones that are of primary interest as they dominate the others) stay at a high level. However, specifically for $m = 1.7$ the loss in quality is more apparent, possibly caused by the too fuzzy assignment of both objects to clusters and clusters to universes.

These experiments show that the presented algorithm is an appropriate approach for clustering in parallel universes, resulting in a clustering that does detect the underlying structure better than individual universe based approaches. We can conclude that for this data set, a small fuzzifier for the object assignment to universes m ($1.1 \leq m \leq 1.3$) and a value of, e.g. $m' = 2$ return a good clustering result given the original class

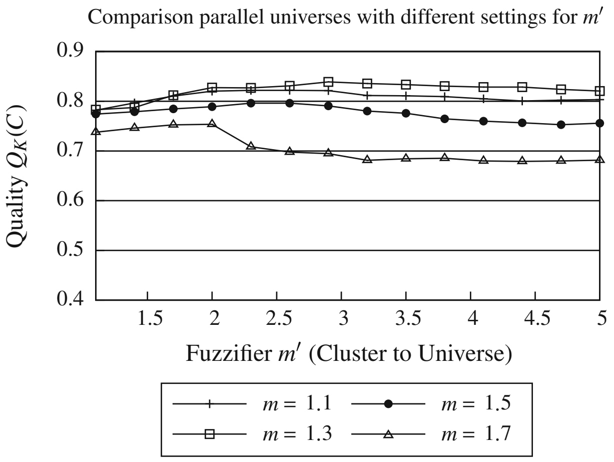
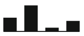
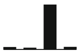
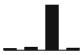
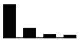
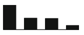
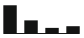
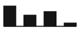
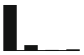
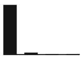
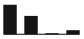
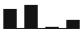
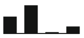
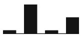
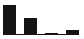
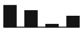
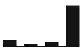
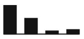
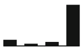


Fig. 4 Quality values for parallel universe approach with increasing fuzziness for cluster to universe assignment

assignment as ground truth. It is less crucial to adjust the parameter controlling the cluster to universe memberships (m'). We showed that when reasonable choices are made for both parameters, clustering in parallel universes is better than single universe clustering.

In order to demonstrate the interpretability of the clusters derived by the presented approach we now concentrate on a clustering with a cluster count $K = 30$ (as above), $m = 1.1$ and $m' = 2$. The clusters covering more than 8 objects are shown in Table 1 along with the cluster size and the normalized entropy of the respective cluster. If a cluster only contains objects of one class (such as cluster 4 or 6) the entropy is 0; larger entropy values indicate mixed clusters. Additionally we also show a small histogram

Table 1 The 18 (out of 30) largest clusters

Cluster	Entropy	Size	Distribution $z_{k,u}$	Covered classes
1	0.061	24		23 × car, 1 ×couch
2	0.065	22		Human, 1 × fish
3	0.216	20		Mostly human, Fig. 5
4	0	19		Plane
5	0.311	16		Plane (two different subclasses)
6	0	15		Human
7	0.241	14		Plane (two different subclasses, Fig. 6)
8	0	13		Sword
9	0	12		Sword
10	0.101	12		Missile, 1× fish
11	0.115	10		Bottles, 1× fish
12	0.115	10		Race cars, 1× weed
13	0	9		Couches
14	0.406	9		4 different classes, mostly missile
15	0	8		Chairs
16	0.133	8		Weed, 1 × planted flower
17	0.344	8		Human, plane, fish
18	0.428	8		Trees, planted flowers, weed

Shown are: the normalized entropy, size, the distribution of the membership values $z_{k,u}$ (form left to right: DBF, SIL, SSD, VOX) and a summary of the contained classes indicating the distribution of the membership values $z_{k,u}$. The four bars represent the memberships to the different universes, namely (form left to right): DBF, SIL, SSD and VOX. The distribution of cluster 3, for example, shows a very high value of the third bar, which translates into a high membership of the cluster to the SSD universe. The objects in clusters 3 and 7 are shown below as examples.

Objects that are covered by cluster 3 are shown in Fig. 5. This cluster has a high membership to the shape-based SSD universe and mostly contains objects from the human class. There is one more very pure cluster of this class in the same universe (cluster 2). We notice that in the SSD universe this particular class is very nicely separated from the remainder. We did not observe such a clear separation for any other class in this universe. This also corresponds to the findings of Bustos et al. (2004).

The other example in Fig. 6 shows a cluster of airplanes, which has an overlap in the image-based universes DBF and SIL and the shape-based descriptor SSD. This cluster has a comparably high entropy, which indicates that it covers different types of objects. It turns out that all objects are airplanes, however, this type of object was further subdivided into three different plane classes, resulting in this artificially increased entropy value.

There are more interesting clusters in this data set, for instance larger groups of swords, which cluster well in the DBF universe (probably due to their very stretched design), or a group of weeds, which only clusters in the volume-based descriptor space. Due to space constraints we have not shown these clusters in separate figures. However, this demonstrates nicely that depending on the type of object certain descriptors, i.e. universes, are better suited to group them. Subspace based or classical multi-view learning methods, which assume identical information in the different input spaces,

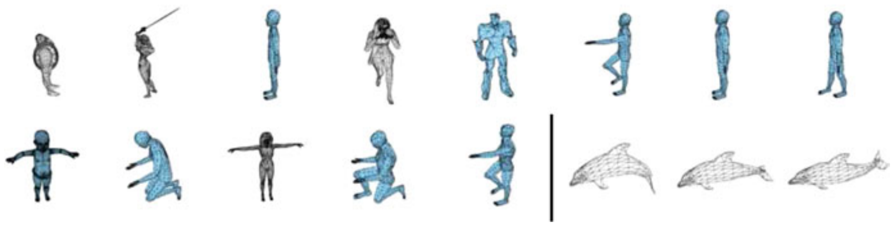


Fig. 5 One of two clusters (#3) primarily in the SSD universe mainly covering the class representing humans. This class in particular separates nicely in this shape-based universe even when the human figures take different poses. We also show the three conflict objects in the cluster on the bottom right

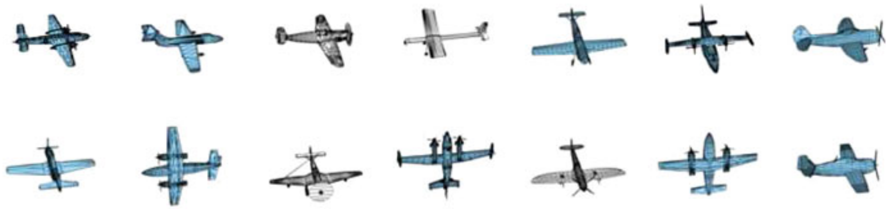


Fig. 6 Cluster #7 spanning both image-based and shape-based descriptor, covering airplanes. Note that they have been categorized into different types of plane classes, which has led to a rather high entropy of this (obviously quite uniform) cluster

would not be able to find these local models in independent universes—combined in a global model—to explain the entire object space. These results demonstrate clearly that learning in parallel universes is a powerful and well suited concept when dealing with these types of information.

7 Summary

We have presented and motivated methods for learning in parallel universes, i.e. for the simultaneous analysis of multiple descriptor spaces. We specifically focused on extensions of a family of fuzzy clustering algorithms. In the classical setup these algorithms minimize an objective function representing a weighted sum between object representations and their (partially) associated cluster prototypes. Using additional terms in the objective function that represent memberships of clusters to universes, these algorithms can also be applied to problems given in parallel universes. The new membership values are then optimized as part of the usual minimization scheme and can later on be interpreted as, for instance, the typicality of a cluster belonging to a certain universe. We compared the presented approach with our previous work, which uses object instead of cluster memberships and also to clustering methods using individual or joint universes. We showed that when using cluster memberships as discussed in this article, the algorithm is better suited for real-world problems as it establishes a connection between different universes and allows for interpretable results.

We applied this extension to the unsupervised analysis of 3D objects. This produced a very intuitive clustering, which could not be detected with other methods. Future research will mainly focus on the normalization of different distance domains. Since all distance functions contribute to the same objective function, the algorithm will prefer universes that produce smaller distance values, although these universes have no inherent advantages other than their small distance range. In the case of the 3D object data the maximum distance in a universe has proven to be a good normalization parameter, however this may not hold for other types of data.

Acknowledgments This work was supported by the German Research Foundation (DFG), BE 1740/12-1. We would also like to thank the reviewers and editors of this special issue for their constructive feedback.

References

3D Benchmark (2008) Konstanz 3D model search engine. http://merkur01.inf.uni-konstanz.de/CCCC/. Accessed 20 March 2009

Aggarwal CC, Yu PS (2000) Finding generalized projected clusters in high dimensional spaces. In: Chen W, Naughton JF, Bernstein PA (eds) Proceedings of the 2000 ACM SIGMOD international conference on management of data, May 16–18, 2000, Dallas, Texas, USA, ACM, pp 70–81

Agrawal R, Gehrke J, Gunopulos D, Raghavan P (1998) Automatic subspace clustering of high dimensional data for data mining applications. In: Proceedings of the 1998 ACM SIGMOD international conference on management of data, ACM Press, New York, NY, USA, pp 94–105

Aggarwal CC, Wolf JL, Yu PS, Procopiuc C, Park JS (1999) Fast algorithms for projected clustering. In: Proceedings of ACM SIGMOD international conference on management of data, pp 61–72

Alqadah F, Bhatnagar R (2008) An effective algorithm for mining 3-clusters in vertically partitioned data. In: CIKM '08: proceeding of the 17th ACM conference on information and knowledge management, ACM, New York, NY, USA, pp 1103–1112. http://doi.acm.org/10.1145/1458082.1458228

Bender A, Glen RC (2004) Molecular similarity: a key technique in molecular informatics. Org Biomol Chem 2(22):3204–3218

Bezdek JC (1981) Pattern recognition with fuzzy objective function algorithms. Plenum Press, New York

Bickel S, Scheffer T (2004) Multi-view clustering. In: Proceedings of the fourth IEEE international conference on data mining (ICDM'04), pp 19–26

Blum A, Mitchell T (1998) Combining labeled and unlabeled data with co-training. In: Proceedings of the eleventh annual conference on computational learning theory (COLT'98), ACM Press, pp 92–100

Bustos B, Keim DA, Saupe D, Schreck T, Vranic DV (2004) Using entropy impurity for improved 3D object similarity search. In: Proceedings of IEEE international conference on multimedia and expo (ICME'2004), pp 1303–1306

Bustos B, Keim DA, Saupe D, Schreck T, Vranić DV (2005) Feature-based similarity search in 3D object databases. ACM Comput Surv (CSUR) 37(4):345–387

Bustos B, Keim DA, Saupe D, Schreck T, Vranić DV (2006) An experimental effectiveness comparison of methods for 3D similarity search. Special issue on multimedia contents and management in digital libraries. Int J Digit Libr 6(1):39–54

Dasgupta S, Littman ML, McAllester DA (2001) PAC generalization bounds for co-training. In: NIPS, pp 375–382

Friedman JH, Meulmany JJ (2004) Clustering objects on subsets of attributes. J R Stat Soc 66(4):815–849

Kailing K, Kriegel HP, Pryakhin A, Schubert M (2004) Clustering multi-represented objects with noise. In: PAKDD, pp 394–403

Klawonn F (2004) Fuzzy clustering: insights and a new approach. Mathware Soft Comput 11:125–142

Kriegel HP, Kröger P, Zimek A (2009) Clustering high-dimensional data: a survey on subspace clustering, pattern-based clustering, and correlation clustering. ACM Trans Knowl Discov Data (TKDD) 3(1):1–58

Krishnapuram R, Keller JM (1993) A possibilistic approach to clustering. IEEE Trans Fuzzy Syst 1(2):98–110

Krishnapuram R, Keller JM (1996) The possibilistic c-means algorithm: insights and recommendations. IEEE Trans Fuzzy Syst 4:385–393

Patterson DE, Berthold MR (2001) Clustering in parallel universes. In: IEEE conference on systems, man and cybernetics, IEEE Press

Rüping S, Scheffer T (eds) (2005) Proceedings of the ICML 2005 workshop on learning with multiple views. http://www-ai.cs.uni-dortmund.de/MULTIVIEW2005/MultipleViews.pdf. Accessed 7 March 2010

Wiswedel B, Berthold MR (2007) Fuzzy clustering in parallel universes. Int J Approx Reason 45(3):439–454