Multi-Aspect Tagging for Collaborative Structuring

Katharina Morik and Michael Wurst

University of Dortmund, Department of Computer Science
Baroperstr. 301, 44221 Dortmund, Germany
morik@ls8.cs.uni-dortmund

Abstract. Local tag structures have become frequent through Web 2.0: Users "tag" their data without specifying the underlying semantics. Every user annotates items in an individual way using the own labels. Even if two users happen to use a tag with the same name, it need not mean the same. Moreover, within the collection of a single user, media items are tagged multiply using different aspects, e.g., topic, genre, occasion, mood. Again, several users applying the same name for an aspect does not imply that actually the same aspect is meant.
Nevertheless, users could benefit from the tagging work of others (*folksonomies*). The set of items clustered together by the same label in one user's collection form a pattern. Knowing this pattern is informative for another user. In contrast to other cluster ensemble methods or distributed clustering, a global model (consensus) is not the aim. Each user wants to keep the tags already annotated, wants to keep the diverse aspects under which the items were organized, and only wishes to enhance the own structure by those of others. A clustering algorithm which structures items has to take into account the local, multi-aspect nature of the task structures. The LACE algorithm [9] is such a clustering algorithm.

1 Local Patterns in Parallel Universes

Large collections of documents, music, and videos are stored today at personal computers. The collections can be structured by a general scheme as is done, e.g., by iTunes, where artist, album, genre, date annotate the music collection. The semantic web has focused on structuring text collections using general ontologies. Users additionally structure their collections concerning several aspects of personal concern. The broad general schemes are enhanced by personal, more specific aspects, e.g., mood, time of day for structuring music, or more finely grained topic structures for text collections. These enhancements are *local*, i.e. they are the personal view of a certain user who does not aim at a global structure for all users. We might call the sets of (media) items which are clustered together by a user (i.e. they are tagged by the same label), a *local pattern.* All these patterns of a user are structured by aspects to form this user's *universe.* Since many users build-up their universes in parallel, there exist many *parallel universes.*

While users tend to start the organization of their personal collection eagerly, they often end up with a large set of items which are not yet annotated and a structure which is too coarse. Facing this situation, we ask, how machine learning techniques could help the users.

If there are enough annotated items, *classification learning* on one user's collection can help. It delivers a decision function φ which maps items x of the domain X to a class g in a set of classes G. New items will be classified as soon as they come in and the user has no burden of annotation any more. However, classification does not refine the structure.

Classification: The input is $I = \{\varphi : S \to G\}$, where $S \subseteq X$ represents the training examples and φ their mapping to the set of predefined classes G. The tasks is to output exactly one function $O = \{\varphi : X \to G\}$ that is able to assign all possible objects to exactly one class in G. The classification setting can be extended to hierarchical classification with $I = \{\varphi : S \to 2^G\}$ and $O = \{\varphi : X \to 2^G\}$.

If the classification is trained on the local patterns of one user, it delivers a *local model* for each local pattern.

Exploiting other users' local models could be performed by *classifier ensembles.* In this case, all the parallel universes together classify new items in a consensus model. For our application, this has several disadvantages. First, the consensus model destroys the specific, individual structure of a user's collection, in the long run. Second, the structure is not refined, because the classes are predefined.

Classifier Ensembles: The input is now a set of mappings $I \subseteq \{\varphi | \varphi : S \to G\}$, but the output is still a single function $O = \{\varphi : X \to G\}$. Again, the setting can be extended to hierarchical classifier ensembles with $I \subseteq \{\varphi | \varphi : S \to 2^G\}$ and $O = \{\varphi : X \to 2^G\}$.

If there is no structure given yet, *clustering* is the method to choose. It creates a structure of groups G for the not yet annotated items $S \subseteq X$. However, it does not take into account the structure which the user already has built up. Semi-supervised clustering obeys given groupings [3]. Note, that clustering does not predict previously unseen items as does classification. Hence, the domain of the function φ is not X but S. In supervised clustering, the input function constrains the output function, where both are defined on the same domain. However, supervised clustering does not refine structures.

Supervised Clustering: In contrast to traditional clustering, a mapping for some objects $x \in S \subseteq X$ to their clusters $g \in G$ is input, $I = \{\varphi : S \to G\}$. The output is $O = \{\varphi : S \to G\}$ or for the hierarchical case $O = \{\varphi : S \to 2^G\}$.

Supervised clustering may deliver a local model for a user's local patterns as well with as without any support from other users.

We may consider the structuring achieved so far a set of partitionings φ_i, each mapping a subset of the given items to a set of groups G_i. For instance, a mapping φ_1 concerning moods and another one, φ_2, concerning time of day may be given. Ensemble clustering then produces a consensus φ_3 which combines these input partitionings [6]. This is almost what we need. However, it does not take into account the reason for applying clustering, i.e. the set S of objects to be integrated. Moreover, it may change the already designed partitionings of the user, which she doesn't want.

Ensemble clustering: For cluster ensembles, a set $I \subseteq \{\varphi_i | \varphi_i : S \rightarrow G_i\}$ of partitions of the objects in S is given. The output is a single partition $O = \{\varphi : S \rightarrow G\}$. Hierarchical cluster ensembles could be defined similarly with $I \subseteq \{\varphi_i | \varphi_i : S \rightarrow 2^{G_i}\}$ and $O = \{\varphi : S \rightarrow 2^G\}$.

We may also consider structures of several users who interact in a network, each offering a clustering $\varphi_i : S_i \rightarrow G_i$. A user with the problem of structuring her left-over items S might now exploit the cluster models of other users in order to enhance the own structure. Distributed clustering learns a global model integrating the various local ones [4]. However, this global consensus model again destroys the structure already created by the user and does not focus on the set S of not appropriately structured items.

Whether own partial clusterings according to different aspects or those of other peers in a network are given, the situation is the same: current clustering methods deliver a consensus model overwriting the given ones and do not take into account S. In addition, users might want to select among proposed models which the learner delivers. Hence, the practical need of the user in organizing her media collection is not yet covered by existing methods. The situation we are facing is actually a new learning task.

Let X denote the set of all possible objects. A function $\varphi : S \rightarrow G$ is a function that maps objects $S \subseteq X$ to a (finite) set G of groups. We denote the domain of a function φ with D_φ. In cases where we have to deal with overlapping and hierarchical groups, we denote the set of groups as 2^G.

Definition 1 (Localized Alternative Cluster Ensembles) *Given a set $S \subseteq X$, a set of input functions $I \subseteq \{\varphi_i : S_i \rightarrow G_i\}$, and a quality function*

$$q : 2^{\Phi} \times 2^{\Phi} \times 2^S \rightarrow R \tag{1}$$

with R being partially ordered[1] LOCALIZED ALTERNATIVE CLUSTERING ENSEMBLES *delivers the output functions $O \subseteq \{\varphi_i | \varphi_i : S_i \rightarrow G_i\}$ so that $q(I, O, S)$ is maximized and for each $\varphi_i \in O$ it holds that $S \subseteq D_{\varphi_i}$.*

Note that in contrast to cluster ensembles, the input clusterings can be defined on any subset S_i of X. Since for all $\varphi_i \in O$ it must hold that $S \subseteq D_{\varphi_i}$, all output clusterings must at least cover the items in S. In [9], we present LACE, a method deriving a new clustering from existing ones by extending the existing clusterings and combining them such, that each of them covers a subset of objects in S.

[1] For example, $R = \mathbb{R}$ if one is interested in a unique solution.

First, we search for a function φ_i in I that best fits the set of query objects S. For all objects not sufficiently covered by φ_i, we search for another function in I that fits the remaining points. This process continues until either all objects are sufficiently covered, a maximal number of steps is reached, or there are no input functions left that could cover the remaining objects. All data points that could not be covered are assigned to the input function φ_j containing the object which is closest to the one to be covered. Alternative clusterings are produced by performing this procedure several times, such that each input function is used at most once.

The LACE algorithm is well suited for distributed scenarios. We assume a set of nodes connected over an arbitrary communication network. Each node has one or several functions φ_i together with the sets S_i. If a node A has a set of objects S to be clustered, it queries the other nodes and these respond with a set of functions. The answers of the other nodes from the input functions I. A computes the output O for S. A node B being queried uses its own functions φ_i as input and determines the best fitting φ_i for S and send this output back to A. The algorithm is the same for each node. Each node executes the algorithm independently of the other nodes.

Putting it into the terms of parallel universes, the method searches in the universe of local models for one which best structures a set of items and integrates this with the most similar local model.

Definition 2 (Learning with Parallel Universes) *In general, different* UNIVERSES *provide a learning algorithm with different informations about a set of items, where it is not necessary that each universe covers the overall set of items or uses the same set of features describing the items. Note, that this definition does not demand the learning algorithm to come to a consensus model.*

Under the heading of clustering with background knowledge, this has shown advantages, e.g., in text clustering [5]. The methods differ in the kind of knowledge they exploit. In our case, we use a whole set of existing cluster models directly to recommend new clusterings instead of invoking a feature based clustering algorithm. Thus our approach preserves a maximum of structure in the users' taxonomies. This is crucial, as the resulting clusters still contain the original node relation and label information making them intuitively comprehensible and sound.

Parallel universes are also related to approaches of multi-view learning and clustering [2,1]. As in multi-view learning, the results of one learning process (here: a user-made cluster model) is exploited to turn an unsupervised learning task into a supervised one.

Connected to this approach is the combination of several partitions into an optimal one [8,6]. For our application, these approaches are not appropriate, since they assume that all cluster models cover the whole set of items. Furthermore, they are not directly applicable to hierarchical structures.

Finally, parallel universes are loosely related to work in the area of meta learning [7]. Meta learning focuses on the question how interactions between similar learning tasks can be exploited to improve the performance of the individual learners. In this sense, our approach can be seen as a meta learning approach to clustering.

In our approach, we have considered a user owning a universe of local models, one for each aspect. A user benefits from the corresponding local models of other universes. The correspondence is established by extensional similarity. We might also identify an aspect with a universe. This would mean that all local models referring to the same aspect form a universe, the diverse aspects are then parallel universes.

References

1. Steffen Bickel and Tobias Scheffer. Multi-view clustering. In *Proceedings of the IEEE International Conference on Data Mining*, 2004.
2. A. Blum and T. Mitchell. Combining labeled and unlabeled data with co-training. In *Annual Conference on Computational Learning Theory (COLT-98)*, 1998.
3. David Cohn, Rich Caruana, and Andrew McCallum. Semi-supervised clustering with user feedback. Technical Report TR2003-1892, Cornell University, 2000.
4. Souptik Datta, Kanishka Bhaduri, Chris Giannella, Ran Wolff, and Hillol Kargupta. Distributed data mining in peer-to-peer networks. *IEEE Inteternet Computing*, special issue on distributed data mining, 2005.
5. Andreas Hotho, Steffen Staab, and Gerd Stumme. Ontologies improve text document clustering. In *ICDM*, pages 541–544, 2003.
6. Alexander Strehl and Joydeep Ghosh. Cluster ensembles – a knowledge reuse framework for combining partitionings. In *Proceedings of AAAI 2002, Edmonton, Canada*, 2002.
7. S. Thrun and J. O'Sullivan. Discovering structure in multiple learning tasks: The TC algorithm. In L. Saitta, editor, *Proc. of the ICML*, San Mateo, CA, 1996. Morgen Kaufmann.
8. Alexander P. Topchy, Anil K. Jain, and William F. Punch. Combining multiple weak clusterings. In *ICDM*, pages 331–338, 2003.
9. Michael Wurst, Katharina Morik, and Ingo Mierswa. Localized alternative cluster ensembles for collaborative structuring. In T. Scheffer et al., editor, *Proc. of the European Conference on Machine Learning (ECML 2006)*, pages 485–496. Springer, 2006.