Deliberation Clustering: an Approach to Improving Clustering Stability and Accuracy

How do you win the Netflix Prize or a KDD Cup, or succeed in any other *supervised* learning task? The state-of-the-art approach is to build an ensemble of most powerful classification methods, train each of them on your data, and then construct an aggregated classifier by carefully combining and weighting the ensemble participants.

One would imagine that an analogous scheme would work in *clustering:* build an ensemble of powerful clustering methods, apply each of them to your data, and then somehow average the results. Well, such a scheme was invented long ago, under the name of Consensus Clustering, or Cluster Ensembles. The only problem is that the consensus clustering does not achieve very impressive results after all. At least, it is not as unbeatable as classifier ensembles in supervised learning. Why is that so?

The problem is that different clustering methods, while being completely unsupervised, construct *very* different models (i.e. clusterings) of the data. Let us say that our data is a collection of geometric shapes (e.g., circles, triangles, rectangles etc.), each of a different color. One clustering algorithm may cluster our data instances according to their shape, another may cluster them according to their color. Averaging those results would be hopeless.

In practice, it is hard to find a clustering method that would cluster data according to any specific criterion (i.e. shape or color), unless it is explicitly incorporated into the optimization objective. A typical clustering method would construct a model that provides some signal *both* about the instances' shapes and about their colors. Averaging such models might not be hopeless, but still not extremely advantageous, as the original models are so dramatically different from each other.

We propose to apply a special type of optimization process (called *Deliberation*) to the original models, which would bring them "closer" to each other, without compromising their quality. Once the models are substantially close to each other, averaging them makes much more sense. Consider the space of all possible clusterings of a dataset (for simplicity, we deal only with *hard* clusterings, i.e. data partitionings). Two clusterings are neighbors in this (discrete) space if one clustering can be transformed into the other by moving *one* data instance from a cluster to another cluster (i.e. the clusterings are identical with the exception of one data instance). By a series of such moves, *any* clustering can be transformed into *any* clustering, such that the task of bringing clusterings closer to each other can be trivially solved. However, we do not want to sacrifice the quality of the original clusterings. Since they were built by trustable clustering algorithms, we can assume that they are pure enough - we are interested in preserving (and further improving) this purity.

Our idea is to bring the clusterings closer to each other by applying to them the same optimization procedure as used in one of the trustable clustering algorithms, while conditioning *all* clusterings on each other. We can illustrate this process on a specific type of clustering algorithms: the *co-clustering* algorithms. The goal of co-clustering is to build a clustering of data instances *X* while *simultaneously* building a clustering of their features *Y*. A typical co-clustering algorithm alternates between optimizing *X* and *Y*. First, it fixes a clustering of *Y* and optimizes clusters of *X* while conditioning them on the (fixed) clustering of *Y*. Next, it fixes the clustering of *X*, and optimizes clusters of *Y* while conditioning them on the fixed clustering of *X* - and so forth in a round-robin fashion.

Our deliberation process starts with *m* clusterings of *X* and *m* clusterings of *Y*. First, it fixes all the *Y* clusterings and applies a clustering optimization procedure to *each* *X* clustering while conditioning it on *all* the (fixed) *Y* clusterings. Next, it fixes the (new) *X* clusterings and applies a clustering optimization procedure to each *Y* clustering while conditioning it on all the fixed *X* clusterings - and so on. This way, all the *X* clusterings get closer to each other, and so do all the *Y* clusterings - we say that the clustering *stability* is improved. Also, since the deliberation process employs a practically unchanged version of the clustering optimization procedure, we can expect that the clustering quality would not decrease (and even improve). Currently, the deliberation process works only on co-clustering methods, however, it can be generalized to any other types of clustering methods.

The rationale behind the deliberation process comes from observing well established decision-making processes in the human society. For example, in jury courts, each juror first attends the hearings and mentally builds his or her model of the case (which is analogous to a clustering algorithm building its local model of the data). Next, the jurors retire for deliberation, where they bring their local models closer to each other by discussing their views and opinions. Only after the deliberation is over, the jurors vote (which corresponds to averaging the clustering results in a consensus clustering scheme). This clear analogy reveals the lack of "deliberation" in the (standard) consensus clustering. Our approach fills this gap.

Consensus clustering with the deliberation phase works really well in practice!!! We applied this method to a variety of datasets, one of which is the popular benchmark 20 Newsgroups (20NG) dataset. When a clustering algorithm of decent quality is applied to the (entire) 20NG dataset, the resulting clustering accuracy is usually less than 0.5. A very good clustering algorithm (such as CLUTO) can achieve around 0.6 clustering accuracy on 20NG. In 2004 I obtained 0.7 clustering accuracy, and since then, for almost 5 years, I have not been able to improve this result even by 0.01. Consensus clustering with deliberation achieves 0.75 - 0.78 clustering accuracy (depending on specific parameter settings). This is an impressive result for a completely unsupervised method on a relatively large dataset of 19,997 instances.

Publication:

R. Bekkerman, M. Scholz, and K. Viswanathan. **Improving Clustering Stability with Combinatorial MRFs.** In *Proceedings of **KDD 2009*** bibtex