# Mixture Model Project

Mathieu Pont and Lucas Rodrigues Pereira

November 2019

**Abstract**

We show through many experiments that consensus clustering can generally significantly improves the results, in term of NMI and ARI, over a simple clustering.

## 1 Introduction

In the context of the Master's Degree on Machine Learning for Data Science, at the University of Paris (Descartes), we have been given the task to discuss Ensemble learning, as part of the "Mixture Model" course.

The idea of this method is to apply many different clustering methods and combine the results to achieve better classification. It involves two steps: Generation and Consensus. First, clusters are generated using multiple approaches; then, these labels compose a new matrix in which a new method is applied (directly or indirectly on this matrix) to form a consensus over the different individual results. Many methods exist and a review can be found in [Vega-Pons and Ruiz-Shulcloper, 2011]. Some of them will be briefly explained here.

In the first part we study the consensus clustering based on a Gaussian Mixture Model as its generation mechanism and a clustering-based method for the consensus.

In the second part we used co-clustering algorithms for the generation process and a consensus with two methods, based on hyper-graphs and co-association matrix.

Finally, in the last part, we generate partitions using von-Mises Fisher Mixture Model and use a consensus identical to the one in the first part.

## 2 Gaussian Mixture Model-based consensus

In this section, we use a Gaussian Mixture Model to generate clusters over three among the five given datasets. These datasets contains flattened images of face expressions (JAFFE) and handwritten digits (OPTDIGITS and MFEA). The number of samples, features and classes are described below:

| Dataset | # samples | # features | # classes |
|---|---|---|---|
| JAFFE | 213 | 676 | 10 |
| OPTIDIGITS | 5620 | 64 | 10 |
| MFEA | 2000 | 240 | 10 |
| MNIST5 | 3495 | 784 | 10 |
| USPS | 9298 | 256 | 10 |

Table 1: Description of datasets.

The two datasets MNIST5 and USPS have been excluded from this study as their high dimensions. Moreover, the libraries used did not work well with these datasets (computation time were too long).

Using the EM-based library *Mclust* (for R), which tests many geometric characteristics (i.e. models) for the Gaussian Distribution, we compare the best clustering approach based on the Bayesian Information Criterion (BIC) to a consensus among all tested approaches. The consensus is made using a Multinomial Mixture Model, as the matrix of the labels of each test is a new categorical dataset.

| Dataset | | Best Model | Consensus |
|---|---|---|---|
| JAFFE | NMI | 0.95 | **0.96** |
| | ARI | 0.92 | **0.93** |
| OPTIDIGITS | NMI | 0.74 | 0.74 |
| | ARI | 0.66 | 0.66 |
| MFEA | NMI | 0.01 | **0.74** |
| | ARI | 2e-05 | **0.66** |

Table 2: Metrics for the best EM partition and for the consensus.

| Dataset | # partitions | Models |
|---|---|---|
| JAFFE | 4 | EII, VII, EEI, VEI |
| OPTIDIGITS | 2 | EII, VII |
| MFEA | 4 | EII, EEI, EEE, EEV |

Table 3: Number of partitions made for each dataset.

We can see from table 2 that the consensus slightly improves the best individual model for JAFFE. However, the OPTIDIGITS dataset partitioning quality has virtually not increased, as the difference was only on the fourth decimal case. This can be explained by the fact that only two models have been used to produce the consensus, as seen in table 3.

The impact of consensus is significant for the MFEA dataset, indeed, the best model (chosen according the BIC criterion) have very poor results whereas the consensus have very good values for the metrics. It shows us that the BIC criterion doesn't always ensure a good clustering, it makes sense since its computation doesn't take into account the labels.

The better the quality of the the partitions used, the more accurate will the consensus. Thus, we can improve the quality of the results by selecting only the best partitions instead of all of them. However, we need a criterion to choose the best ones. Since Mixture Models maximize the likelihood (directly or indirectly) we can take the partitions with higher likelihood. We can also use the BIC and AIC criterion to choose the best partitions. The partitions having the higher value among these three criteria does not necessarily have the highest NMI or ARI (as seen for MFEA dataset) but they can nevertheless be considered a good choice to select some partitions. Another way could be a "brute force" way where we try many combinations of partitions to find the best combination, since they are all already generated it doesn't involves computation time cost.

We tried to select best partitions using this method but the metrics didn't increase. We must notice that we have not very much partitions for each dataset (a maximum of 4), so we don't have many choices in the selection.

# 3 Consensus with Co-Clustering

In this section we study the impact of co-clustering (simultaneously clustering variables and observations) on the consensus learning. Three different algorithm have been used: Coclust Info (Information-Theoretic), Coclust SpecMod (Spectral approximation of the modularity), Coclust Mod (Graph modularity). All these methods are available in the *Coclust* package for Python. We used Google Colaboratory to run the experiments.

Five text datasets (document-term datasets) have been selected. They are all normalized using TF-IDF and L2-norm.

| Dataset | # samples | # features | # classes |
|---|---|---|---|
| CLASSIC4 | 7094 | 5896 | 4 |
| CSTR | 475 | 1000 | 4 |
| RCV4 | 6387 | 16921 | 4 |
| NG5 | 4905 | 10167 | 5 |
| NG20 | 18846 | 14390 | 20 |

Table 4: Description of datasets.

The idea is to find a consensus out of the top 10 clusterings achieved by 200 different initializations. Two different approaches have been used to find the consensus:

- **Hypergraph consensus clustering**: proposed by [Strehl and Ghosh, 2003] this method consists on finding a new partition that share the higher NMI with all the proposed partitions. The problem is defined as a (hyper)graph partitioning problem.
- **Clustering on the co-association matrix**: this matrix indicates how many times a sample appears in both clusters among the different partitions, then we can use a clustering algorithm on this matrix to make the consensus. Here, the same method used to generate the partitions is used to do clustering on the co-association matrix.

To make these partitions we run the algorithm 200 times and take the 10 best partitions according to the optimized criterion (mutual information or modularity).

Then we make the same experiments adding an extra partition made by Spherical k-means. Since this algorithm is particularly well suited for textual data, it is expected to improve the results.

In addition, we have experimented Coclust Info varying the number of column clusters by multiples of the number of row clusters (2x, 3x), but keeping the same amount of desired rows clusters. These variations are respectively represented as "(col x2)" and "(col x3)" in Table 5.

As regard the methods Coclust Mod and SpecMod, we cannot specify the number of columns cluster since they are diagonal co-clustering methods (i.e. same number of rows cluster than columns ones).

Finally, we make one last experiment combining the partitions of Coclust Info, Mod, SpecMod and Spherical k-means. We have selected 3 partitions from each method in order to have a well-balanced representation of each algorithm in this experiment (i.e. approximately same number of partitions for each experiments).

| Generation | Consensus | CLASSIC4 | | CSTR | | RCV4 | | NG5 | | NG20 | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | NMI | ARI | NMI | ARI | NMI | ARI | NMI | ARI | NMI | ARI |
| Info | Hypergraph | 0.75 | 0.73 | 0.75 | 0.81 | 0.53 | 0.45 | 0.74 | 0.69 | 0.58 | 0.43 |
| | Info | 0.75 | 0.72 | 0.79 | 0.81 | 0.52 | 0.45 | 0.75 | 0.69 | 0.60 | 0.43 |
| Info + SkMeans | Hypergraph | 0.74 | 0.70 | 0.78 | 0.83 | 0.52 | 0.45 | 0.74 | 0.69 | 0.62 | 0.47 |
| | Info | 0.75 | 0.72 | 0.74 | 0.70 | 0.52 | **0.54** | 0.79 | 0.80 | 0.61 | 0.44 |
| Info (col x2) | Hypergraph | 0.77 | 0.73 | 0.77 | 0.82 | 0.51 | 0.44 | 0.87 | 0.90 | 0.65 | 0.52 |
| | Info (col x2) | **0.78** | **0.75** | 0.77 | 0.82 | 0.52 | 0.45 | 0.87 | 0.90 | 0.65 | 0.51 |
| Info (col x2) + SkMeans | Hypergraph | 0.76 | 0.72 | 0.77 | 0.81 | 0.51 | 0.44 | 0.87 | 0.89 | 0.65 | 0.53 |
| | Info (col x2) | **0.78** | **0.75** | 0.76 | 0.80 | 0.53 | 0.45 | 0.78 | 0.72 | **0.66** | 0.52 |
| Info (col x3) | Hypergraph | 0.61 | 0.44 | 0.80 | **0.84** | 0.48 | 0.41 | 0.89 | **0.92** | **0.66** | **0.54** |
| | Info (col x3) | 0.77 | 0.72 | 0.80 | **0.84** | 0.49 | 0.43 | 0.89 | **0.92** | 0.65 | 0.50 |
| Info (col x3) + SkMeans | Hypergraph | 0.76 | 0.71 | **0.81** | **0.84** | 0.50 | 0.43 | **0.90** | **0.92** | **0.66** | 0.53 |
| | Info (col x3) | 0.76 | 0.70 | **0.81** | **0.84** | 0.51 | 0.43 | 0.89 | **0.92** | **0.66** | 0.51 |
| SpecMod | Hypergraph | 0.00 | 0.00 | 0.76 | 0.80 | 0.38 | 0.46 | 0.74 | 0.65 | - | - |
| | SpecMod | 0.01 | 0.00 | 0.76 | 0.80 | 0.47 | 0.43 | 0.73 | 0.65 | - | - |
| SpecMod + SkMeans | Hypergraph | 0.27 | 0.1 | 0.76 | 0.80 | 0.43 | 0.45 | 0.72 | 0.66 | - | - |
| | SpecMod | 0.38 | 0.2 | 0.76 | 0.80 | 0.52 | 0.44 | 0.73 | 0.65 | - | - |
| Mod | Hypergraph | 0.72 | 0.68 | 0.77 | 0.81 | 0.53 | 0.46 | 0.83 | 0.85 | 0.48 | 0.29 |
| | Mod | 0.72 | 0.68 | 0.75 | 0.78 | **0.54** | 0.48 | 0.74 | 0.68 | 0.51 | 0.24 |
| Mod + SkMeans | Hypergraph | 0.74 | 0.71 | 0.76 | 0.81 | 0.53 | 0.47 | 0.83 | 0.85 | 0.49 | 0.31 |
| | Mod | 0.72 | 0.68 | 0.78 | 0.82 | **0.54** | 0.48 | 0.84 | 0.86 | 0.54 | 0.26 |
| All | Hypergraph | 0.76 | 0.68 | 0.79 | 0.83 | 0.52 | 0.46 | 0.87 | 0.88 | - | - |
| | Info (col x3) | 0.77 | 0.72 | 0.79 | 0.82 | 0.51 | 0.42 | 0.86 | 0.88 | - | - |
| | SpecMod | 0.77 | 0.72 | 0.80 | 0.83 | 0.53 | 0.45 | 0.87 | 0.88 | - | - |
| | Mod | 0.76 | 0.71 | 0.79 | 0.82 | 0.47 | 0.43 | 0.87 | 0.89 | - | - |
| Info (col x3) | - | 0.74 | 0.70 | 0.74 | 0.77 | 0.49 | 0.43 | 0.84 | 0.87 | 0.60 | 0.46 |

Table 5: Metrics for all the experiments for the consensus on co-clustering partitions. "All" is the partitions made by Info (col x3), SpecMod, Mod and SkMeans. The last row shows the results without consensus.

To assess our experiments we use two well-known metrics for clustering evaluation: the NMI (Normalized Mutual Information) and the ARI (Adjusted Rand Index).

By analysing Table 5, as expected, the addition of the Spherical k-means partition have improved the quality of the results. However, the impact of Spherical k-means on the result is limited and this can be explained by the fact that it is only one "vote" against the other 10 from other partitioning methods.

Moreover, also confirming our expectations, by increasing the number of column clusters, the metrics for Coclust Info have improved. We can see that the best results for each dataset is generally for Coclust Info with the higher number of columns clusters used (three times more than row clusters).

We could have thought that the combination of all algorithms ("All" in Table 5) for the generation process leads to better results. Indeed, since we combine different source, the results could have been better. But it is actually the opposite, the results of Coclust Info (col x3) is generally higher than the others therefore the metrics are pulled down by the partitions of the other algorithms. Even if we can see that they are quite good enough. Coclust SpecMod gives very poor results for CLASSIC4, this need to be investigate, maybe the spectral decomposition gives not a good representation to make a clustering on (for this dataset).

The impact of consensus clustering is more notable here than in the first part. We have run a simple experiment, without consensus, using Coclust Info (col x3) and we see that the metrics are significantly lower than those with the consensus.

For the consensus function we see that, in our case, the hypergraph and the co-association matrix methods give more or less the same results.

# 4 Consensus with von Mises-Fisher Mixture Model

The von Mises-Fisher Mixture Model, based on the von Mises-Fisher distribution, can be interpreted as a generalization of the Spherical k-means, which suits well for textual data. Four datasets have been selected out of the CLUTO package. For obvious practical reasons, we have selected the following four datasets, which have lower dimentionality.

| Dataset | # samples | # features | # classes |
|---|---|---|---|
| FBIS | 2463 | 2000 | 17 |
| RE0 | 1504 | 2886 | 13 |
| TR23 | 204 | 5832 | 6 |
| WAP | 1560 | 8460 | 20 |

Table 6: Description of datasets.

We used the package *movMF* and *skmeans* for R. The first one proposes many different methods for the M-step in the von Mises-Fisher Mixture Model, we used all of them to make our matrix of partition for the consensus and we added an extra partition using Spherical k-means.

We also compare the result to the best Gaussian Mixture Model given by *Mclust*, we expected to find that the results of this model are lower than the von Mises-Fisher one since the latter is more suited to directional data.

| Dataset | | Best GMM | Best vMF | Consensus |
|---|---|---|---|---|
| FBIS | NMI | - | 0.51 | **0.53** |
| | ARI | - | 0.29 | **0.33** |
| RE0 | NMI | 0.14 | 0.35 | 0.35 |
| | ARI | 0.04 | 0.20 | **0.22** |
| TR23 | NMI | 0.03 | 0.20 | **0.27** |
| | ARI | -0.04 | 0.09 | **0.18** |
| WAP | NMI | - | **0.53** | 0.52 |
| | ARI | - | 0.24 | **0.29** |

Table 7: Metrics for the best Gaussian Mixture Model (GMM), for the best vMF (according to likelihood) and for the consensus.

As seen in table 7 we weren't able to run Gaussian Mixture Model for some dataset because it took to much time. For the dataset for which it was possible we had poor results, it can be explained by the fact that GMM doesn't take into account the directionality of the data unlike von Mises-Fisher Model. It's why, as expected, vMF gives better results. Moreover we see that the consensus over all possibles vMF and Spherical k-means is generally better than the others.

It is interesting to notice that TR23 is a very particular dataset with very few samples (204) and many more features (5832) and it's probably why the different models did not have good results. But it is for this dataset that the consensus give most improved metrics.

# 5 Conclusion and Future Works

## 5.1 Conclusion

We seen through this work that consensus clustering can improve results compared to a classical clustering.

We showed that in our case von Mises-Fisher Mixture Model and Spherical k-means are, as expected, much better suited for directional data than Gaussian Mixture Model. We also realized that the latter can need more computational power than a simple k-Means (we weren't able to run GMM for some dataset). This algorithm can therefore be a good choice for a quick clustering even if it's important to keep in mind that GMM can be considered as a generalization of k-Means.

Finally, our github repository is open source and free to use[1].

## 5.2 Future works

It would have been better to run our experiments multiple times and compute the mean and standard deviation of them, the results would be more significant and reliable. However, due to our computation and our time limit, we were not able to do this.

Another interesting thing would have been to vary the number of row cluster in the generation process while keeping the true number of clusters for the consensus function. We can also push further the experiments (but on other dataset) linked to the selection of the best partitions.

# References

[Strehl and Ghosh, 2003] Strehl, A. and Ghosh, J. (2003). Cluster ensembles — a knowledge reuse framework for combining multiple partitions. *J. Mach. Learn. Res.*, 3:583–617.

[Vega-Pons and Ruiz-Shulcloper, 2011] Vega-Pons, S. and Ruiz-Shulcloper, J. (2011). A survey of clustering ensemble algorithms. *IJPRAI*, 25:337–372.

[1] https://github.com/MatPont/Mixture_Model_Consensus