Brain tissue segmentation using Expectation Maximization (EM) algorithm for Gaussian Mixture Models (GMM)

X. Beltran Urbano[1] and F. Hartmann[1]

[1] *University of Girona, Erasmus Mundus Joint Master Degree in Medical Imaging and Applications (MAIA)*

Abstract—Image segmentation remains a challenging task in the medical domain. Probabilistic clustering methods have been and still are frequently used algorithms to facilitate this task. In this report, the Expectation Maximization algorithm is combined with Gaussian Mixture Model to perform the task of brain tissue segmentation. The algorithm has been validated on a multi-modal dataset (n=5) reaching dice scores of 0.85, 0.80 and 0.81 for White Matter, Gray Matter and Cerebrospinal Fluid, respectively. Furthermore, the algorithm is compared to other segmentation tools such as K-Means or Statistical Parametric Mapping (SPM), achieving similar results.

Keywords—Brain tissue segmentation, Expectation Maximization (EM), Gaussian Mixture Models (GMM), MRI, Clustering algorithms, Image processing, Medical imaging

I. Introduction

Image segmentation is one of the oldest approaches in the medical imaging field. The aim of this procedure is to split the image into subregions to improve the diagnosis and treatment of the patients. In brain imaging, a very traditional approach consists of extracting the tissue from a brain image through the process of partitioning into a collection of distinct regions that share common attributes, including texture, intensity, homogeneity, and so forth. The extraction of tissues such as cerebrospinal fluid (CSF), white matter (WM), and gray matter (GM) from magnetic resonance (MR) images is a frequently utilized technique in the field of quantitative brain analysis. By separating normal tissues from brain lesions, diseases such as Alzheimer's disease (AD), Parkinson's disease, and others can be better identified and, hence, treated.

This segmentation can be carried out throughout supervised or unsupervised algorithms. One of the most utilised algorithms for this task is clustering. One clustering method that has been widely used for image segmentation is Expectation Maximization (EM) algorithm, using Gaussian Mixture Models. This statistical technique is utilized to assess the likelihood of each voxel within the image, enabling the assignment of these voxels to the clusters that exhibit the greatest similarity.

In this laboratory, the primary objective is to develop an entire pipeline for brain tissue segmentation utilising EM algorithm and GMM.

II. Dataset

To implement the brain tissue segmentation, we have utilised a multimodal dataset composed by 5 different MRI images. For each of the cases, we have the following modalities:

- T1-weighted (T1)
- FLAIR-weighted (FLAIR)
- Ground Truth (GT)

An example of the different image modalities of the dataset can be observed in the Fig.1.

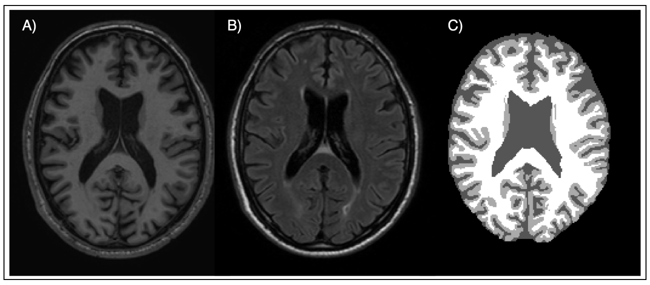

Fig. 1: Example of the different modalities of the dataset. **A)** T1 **B)** FLAIR **C)** GT

III. Method

The Expectation Maximization algorithm consists of two steps: The expectation step used to update membership weights of each data point (each voxel) and the maximization step used to recompute the parameters of the model. The steps will be explained in subsection I and subsection II. First, a finite mixture model with K components can be defined as:

$$p(\underline{x}|\theta) = \sum_{k=1}^{K} \alpha_k \cdot p_k(\underline{x}|\theta_k) \tag{1}$$

Where

$$\begin{aligned} p(\underline{x}|\theta) &: \text{Probability density function} \\ \underline{x} &: \text{d-dimensional data} \\ \theta &: \text{Parameters of the model} \\ \alpha &: \text{Mixture weights} \end{aligned}$$

Under the assumption that $\underline{x}$ follows a Gaussian distribution, $p_k(\underline{x}|\theta_k)$ can be described with a multivariate Gaussian distribution. This results in the following equation:

$$\begin{aligned} p_k(\underline{x}|\theta_k) = & \frac{1}{(2\pi)^{d/2}|\Sigma|^{1/2}} \\ & \exp\left(-\frac{1}{2}(\underline{x}-\underline{\mu_k})^T \Sigma_k{}^{-1}(\underline{x}-\underline{\mu_k})\right) \end{aligned} \quad (2)$$

With

$$\begin{aligned} \theta_k &: \text{representing the mean } \underline{\mu_k} \text{and covariance } \Sigma_k \\ d &: \text{Number of modalities} \end{aligned}$$

Due to computational limitations, the exponent of Equation 2 is not calculated in a matrix form, but in the following form instead:

$$-\frac{1}{2}\sum\left(\left((\underline{x_i}-\underline{\mu})\Sigma^{-1}\right)\odot(\underline{x_i}-\underline{\mu})\right) \quad (3)$$

With $\odot$ being an element-wise multiplication and $\sum$ being the sum over the columns of the resulting matrix.

I. Expectation step

Each point in the vector $\underline{x}$ is assigned a membership weight ω of belonging to a cluster k. The membership weight can be computed using Equation 4.

$$\omega_{i,k} = \frac{p_k(\underline{x}|\theta_k)\cdot\alpha_k}{\sum_{m=1}^{K} p_m(\underline{x}|\theta_m)\cdot\alpha_k} \quad (4)$$

Each point has K membership weights - one for each cluster. The sum of the membership weights of a point is 1. In the last iteration the point is assigned to the component with the highest membership weight.

II. Maximization step

The maximization step updates the parameters of the Gaussian Mixture Model: $\underline{\mu_k}$, Σ_k and α_k. The formulas are computed in consecutive order as described in Equation 5.

$$\begin{aligned} \alpha_k &= \frac{N_k}{N} \\ \underline{\mu_k} &= \frac{1}{N_k}\sum_{i=1}^{N}\omega_{i,k}\cdot\underline{x_i} \\ \Sigma_k &= \frac{1}{N_k}\omega_{i,k}(\underline{x_i}-\underline{\mu_k})(\underline{x_i}-\underline{\mu_k})^T \end{aligned} \quad (5)$$

III. Initialization and convergence

The means are either initialized randomly or with K-means. The covariance is initialized as a random diagonal matrix, which uphold the requirement of a positive semi-definite matrix in Equation 2. In order to check for convergence, the log of Equation 2 is computed and compared with the logarithm of the previous iteration. If the difference is less than a threshold, the algorithm is stopped. The threshold is empirically chosen to be 10^{-6}. If the algorithm is not converging, a maximum number of 500 iterations is used. An illustration of the evolution of the log-likelihood in the EM algorithm is presented in Figure 2. As represented in the image, this value increases throughout execution until it reaches a plateau; at that point, convergence is achieved, and the algorithm stops.

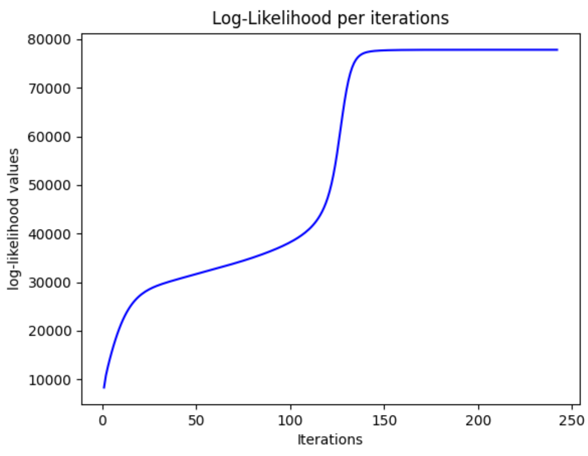

Fig. 2: Evolution of log-likelihood in the EM algorithm.

IV. Labels reassignment

Since the assigned labels are different, they might not match the labels of the ground truth. To ensure comparability, each tissue of the ground truth is compared with every cluster of the segmentation. The comparison is done by dice score. The predicted labels are then changed to match the label of the ground truth with the highest dice score.

IV. Results

I. Metrics

To assess the efficacy of our strategy and evaluate the experiments carried out in the laboratory, the different cases of our dataset were evaluated using the metric *Dice Score(DSC)*, which formula can be shown in the Equation 6:

$$DSC = \frac{2\times TP}{2\times TP+FP+FN} \quad (6)$$

Both the mean and standard deviation were calculated during the tests in order to enhance the available information and afterwards analyse the data in a comprehensive manner.

II. Experiments

In this section, we will present the results obtained from each of the experiments carried out in this laboratory. As mentioned before, this algorithm has been tested with different modalities (T1 and T1+FLAIR) and different initializations (Random and using KMeans). The results of these experiments have been evaluated using DSC and can be seen in the

Algorithm 1 Gaussian Mixture Model

```
1:  procedure GaussianMixtureModel($k, data, maxIterations$)   ▷ Initialization
2:      Initialize $k, data, maxIterations$, and other parameters
3:      Initialize control variables for convergence
4:  end procedure
5:
6:  function Run
7:      while not Converged do
8:          ExpectationStep()
9:          MaximizationStep()
10:         Increment $completedIterations$
11:     end while
12:     return clusterAssignments
13: end function
14:
15: function ExpectationStep
16:     ComputeMembershipWeights()
17: end function
18:
19: function MaximizationStep
20:     UpdateMixtureWeights()
21:     UpdateMeans()
22:     UpdateCovariance()
23: end function
24:
25: function Initialization($initialization\_type$)
26:     if $initialization\_type$ is "Random" then
27:         Initialize randomly
28:     else if $initialization\_type$ is "KMeans" then
29:         Use KMeans for initialization
30:     else
31:         Raise exception
32:     end if
33: end function
34: function IsConverged
35:     Calculate newLog
36:     if convergence criteria met then
37:         return True
38:     else
39:         Update prevLog
40:         return False
41:     end if
42: end function
43:
44: function LogLikelyhood
45:     Calculate log-likelihood
46:     return result
47: end function
```

Table 1. In addition, a visual example of the segmentations obtained with these experiments can be shown in the Figure 6, at the end of this document.

Finally, to better discuss the results obtained with our algorithm, we have also executed, on the same dataset, the segmentation tools KMeans and SPM. The results of this experiment can be seen in Table 2.

V. Discussion

In this section, the results of the Expectation Maximization Algorithm will be discussed and compared to other segmentation tools such as SPM or K-means.

I. Comparison of initialization methods

As explained in section III, the expectation maximization algorithm can be initialized randomly or with the clusters predicted by the K-means algorithm. Table 1 shows the combination of initialization methods and modalities and the resulting dice scores. Additionally, the required iterations and computation time are stated. It can be seen that all combinations of modalities and initialization perform similarly with respect to the dice score except for a random initialization combined with a T1 scan which is performing worse. This might be due to the fact that the algorithm failed to converge for the 5th case, resulting in a dice score of 0.48 for WM and 0.67 for both GM and CSF. Additionally, it can be stated that the use of K-means reduces the number of iterations until convergence by a factor of three, allowing for a faster computation of the segmentation. It is worth noting that the computation time of K-means is included in the measured time.

II. Comparison of Modalities

Both the segmentation with a Gaussian Mixture Model, as shown in Figure 3, and solely with K-means, as shown in Table 2, display the advantage of combining different modalities across all dice score means. Nevertheless, comparing the dices of the T1 modality to the combined modalities, reveal that the improvements are all within the 1σ confidence interval. Therefore, the difference of a combined modality over the T1 modality is statistically insignificant.

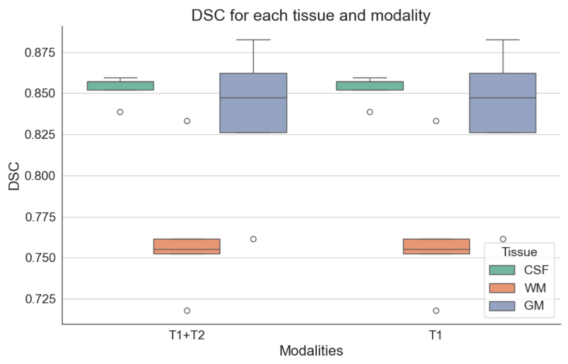

Fig. 3: Comparison of the dice obtained for each of the tissues summing all modality types.

III. Comparison with K-means

Comparing the Expectation Maximization Algorithm with the sole usage of K-means, demonstrates a significant difference in the computation time. While the K-means algorithm segments the scan in under four seconds, the EM-Algorithm takes from twenty-five to one-hundred-fifty seconds. This can be attributed to the implementation and the complexity of the algorithm itself. While K-means only updated the mean in each iteration, the Gaussian mixture model calculates means, co-variances, membership-weights, mixture-

TABLE 1: EVALUATION RESULTS OF THE EXPECTATION MAXIMIZATION ALGORITHM ON THE TEST SET

Initialization	**Modalities**	**GM** [DSC]	**WM** [DSC]	**CSF** [DSC]	**Iterations**	**Time** [s]
KMeans	T1	0.8457 ± 0.0436	0.8015 ± 0.0300	0.8120 ± 0.0251	116 ± 4	**24.36 ± 1.28**
KMeans	T1+FLAIR	0.8457 ± 0.0436	**0.8017 ± 0.0304**	**0.8129 ± 0.0249**	**110 ± 17**	51.48 ± 8.10
Random	T1	0.8025 ± 0.0819	0.7284 ± 0.1383	0.7860 ± 0.0670	346 ± 108	65.87 ± 18.42
Random	T1+FLAIR	**0.8468 ± 0.0450**	0.7978 ± 0.0233	0.8082 ± 0.0278	333 ± 117	147.71 ± 51.16

TABLE 2: EVALUATION RESULTS OF KMEANS AND SPM

Method	**Initialization**	**Modalities**	**GM** [DSC]	**WM** [DSC]	**CSF** [DSC]	**Time** [s]
K-means	K-means ++	T1	0.8222 ± 0.0356	0.7464 ± 0.0338	0.8542 ± 0.0078	3.57 ± 0.12
K-means	K-means ++	T1+FLAIR	**0.8243 ± 0.0357**	**0.7500 ± 0.0315**	**0.8560 ± 0.0071**	**3.61 ± 0.19**
SPM	-	T1	**0.75 ± 0.03**	**0.81 ± 0.03**	**0.76 ± 0.02**	-

weights based on a Gaussian density function. Additionally, K-means is used in the faster initialization to predict the first clusters. Despite this, the performance in dice score of the Gray matter is similar. Furthermore, K-means outperforms the EM-Algorithm in terms of cerebrospinal fluid segmentation. This is clearly visible in Figure 4. However, the dice in the white matter segmentation is significantly better using the EM-Algorithm. This being said, more data is required to allow for a better comparison between both algorithms.

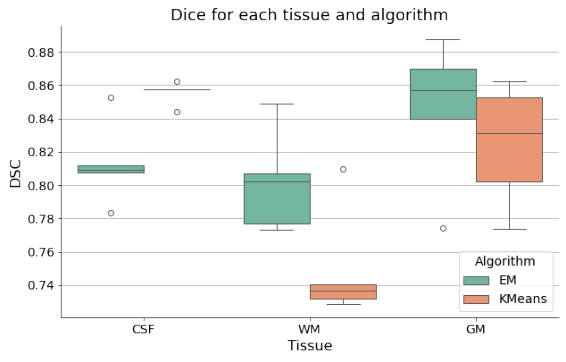

Fig. 4: dice obtained for each of the tissues using T1+FLAIR with EM (Kmeans initialization) and K-means only.

IV. Comparison with SPM

The comparison in Figure 5 suggests that the EM-Algorithm is capable of outperforming SPM using the T1-modality with respect to the dice score. However, the EM-Algorithm uses the ground truth to obtain only the voxels belonging to white matter, gray matter and cerebrospinal fluid. In contrast to SPM, which used the entire scan. This makes the segmentation task significantly harder as SPM has to additionally segment background and skull. A direct comparison with the given results is therefore not possible.

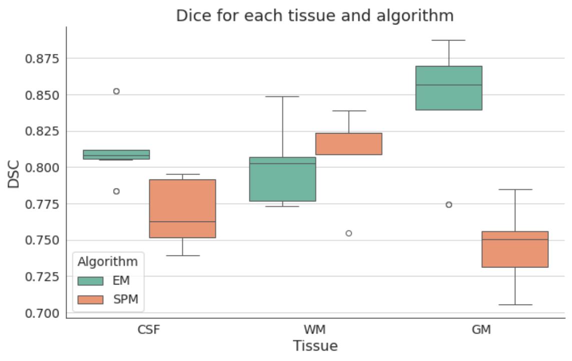

Fig. 5: Example of the dice obtained for each of the tissues and different algorithms (EM and SPM).

VI. CONCLUSION

As we conclude our exploration in this laboratory, several key findings and insights emerge. The marriage of the Expectation Maximization (EM) Algorithm with Gaussian Mixture Models demonstrated its potential in brain tissue segmentation. The robustness and intricacy of this combination were evident, especially when it came to the complex calculations involving means, co-variances, and mixture-weights grounded in a Gaussian density function. Our comparative analysis revealed insightful contrasts, particularly when the EM-GMM approach was compared with KMeans and SPM. Although EM segmented WM and GM more accurately than KMeans, this algorithm demonstrated superior performance in the segmentation of CSF and emphasized speed and effectiveness. The comparison with SPM, on the other hand, underscored the inherent challenges faced by SPM, given it used the entire scan versus the targeted voxel approach by the EM Algorithm. As enthusiastic as we were in undertaking this lab, our excitement only grows further. We eagerly look forward to our next venture – a lab that promises to delve into the integration of atlas information. The journey ahead beckons, and we are poised to explore, learn, and innovate.

VII. DESIGN AND IMPLEMENTATION

In the development of our Gaussian Mixture Model, we adhered strictly to principles of object-oriented programming, crafting our solution from the scratch. The entire implementation was rooted in theoretical concepts discussed during the theory lectures, ensuring a robust foundation based on established academic principles. This methodical approach facilitated a deeper understanding of each component of the algorithm, as we manually pieced together every function in the class. Below, we provide a detailed pseudocode that encapsulates the essence of our implementation, offering insights into the meticulous construction of our Gaussian Mixture Model.

VIII. Project Management

This project, which is a collaborative effort between two team members, was initiated completely from scratch, demanding creative input and exhaustive planning. Initially expected to be completed within the lab hours, the project slightly exceeded this timeframe, requiring an additional two hours of work at home for each member. This extension was critical because it allowed for comprehensive development and refinement beyond the foundational work done during lab sessions. Despite this, our time management strategy was primarily focused on increasing productivity within lab hours by leveraging the benefits of pair programming to expedite task execution and effectively troubleshoot. The journey from a blank slate to a fully-functional application demonstrated our dedication to quality and efficiency, even when it meant exceeding our original time estimates.

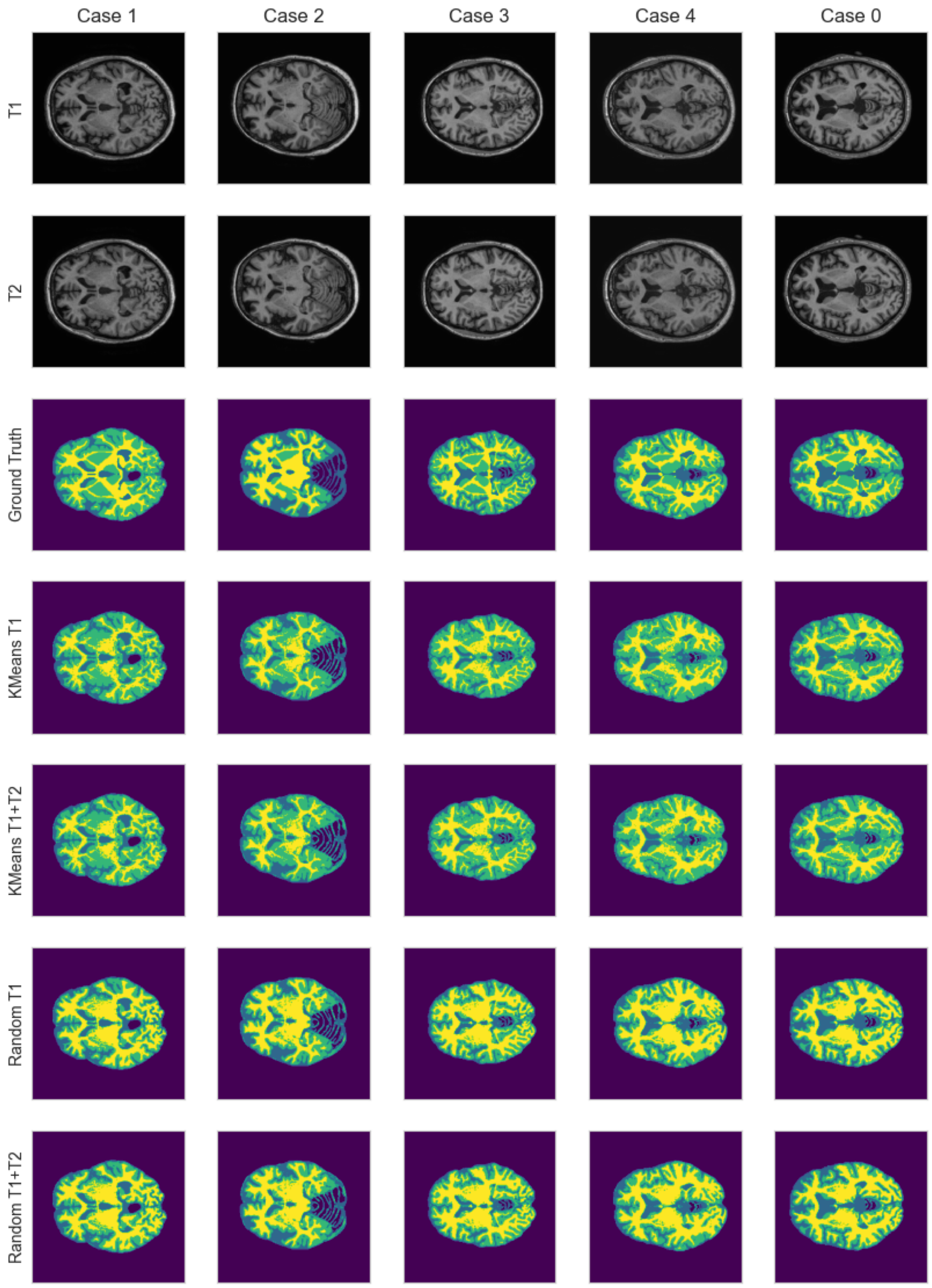

Fig.6: Result of the segmentation using our approach with different initializations and modalities.