University of Illinois Fall 2018

ECE 544NA: Pattern Recognition
Lecture 16: K-Means Clustering (October 18)

Lecturer: Alexander Schwing Scribe: Ahmed Mzid

1 Introduction

Discriminative modeling approaches can solve a variety of problems and have several applications in the field of Machine Learning. However, they depend heavily on the labeling of training data, which can be sometimes time-consuming and very expensive or even not possible for a huge amount of data. Therefore, a second category of modeling approaches has gained a lot of popularity in the recent years because it relies on unsupervised learning methods. These are the generative modeling approaches, and one of the oldest, yet most popular generative approaches is the K-means algorithm used for data clustering.
In this lecture, we start with a brief recapitulation and discussion of the need of generative models. The subsequent section explains the concept of clustering and one example of its applications. Then, the K-means algorithm and some of its properties are introduced. This algorithm has several extensions which will be discussed in the sixth section. Finally, two main applications of the K-means algorithm will be introduced.

2 From Discriminative To Generative Modeling

This first section recapitulates briefly the purpose and high-level approach used in the first part of this class and introduces a second type of modeling approach relying on unsupervised learning methods.

2.1 Recapitulation: Discriminative Modeling

In the first part of this class, the focus was to create a model which predicts an output Y on the basis of some input X and of some model-dependent learned parameters θ. These types of models are called discriminative. Discriminative models learn a probability distribution in the form of $P(Y|X,\theta)$ and are usually applied either to classify the inputs X to some category $Y \in \{1,\dots,C\}$ where C is the number of classes (classification) or to approximate a mapping (or a function) from the inputs X to a real-valued output Y (regression). The main approach consists of training the model usually through the minimization of a specific loss function using some optimization technique and a preferably large dataset. The purpose of the training step is to learn the model-dependent parameters. On the basis of these parameters, the inference is applied to predict the data for testing purposes or for some unlabeled real-world data.

Considering the example in fig. 1, semantic segmentation can be achieved by training a classifier which gives as output for instance just a local evidence. The result of this local evidence can then be used to classify each pixel. A fancier way to do this task would be to apply some structured technique and the benefit would be to consider the neighboring pixels as well and thus generate a joint prediction. In both cases, a huge dataset with labeled images is needed. A labeled image in this context is an image where each pixel is associated to some class or object. This means that a big preprocessing is needed in order to label each pixel in each image in the training dataset. This should in general be performed by human intervention. Therefore, the need for labels for non-obvious tasks is a big time-consuming and expensive problem in the case of training a discriminative model.

Figure 1: Example of a classification task for semantic segmentation

2.2 Generative Modeling

Generative modeling (or unsupervised learning) is applied when only output data are given without any inputs with the aim of discovering how this data is structured or generate similar data. Since no labels are given as in the case of learning for discriminative models, the density estimation is formalized to have the form $P(X|\theta)$ for generative modeling.
The unsupervised learning applied to train generative models is mostly more similar to the way humans learn. Additionally, it is more widely applicable and less expensive since no human intervention is needed for data labeling. One of the most famous examples is data clustering into groups. This will be introduced in the next section.

3 Clustering concept

One of the most basic and simple ways to model data is to make the assumption that it consists of several groups called clusters. A cluster is a set of data points which share some similar properties according to a specific similarity metric. The process of dividing the data points into clusters is called clustering.

The example in the figures below illustrates an instance of clustering applications. The purpose is to separate the flower in the foreground from the black background in the input image (fig. 2). The used approach here is to give each pixel a unique label which defines the category it belongs to. The category in this example can be either background or foreground since we have a single object in the foreground. To achieve this goal, we draw a histogram (fig. 3) which shows the number of pixels as a function of the intensity value ranging from 0 to 255. Note that the feature space in this case is one-dimensional since the input is a grayscale image. If the image contains colors, we will have three channels and our feature space will be three-dimensional. Back to the one-dimensional feature space, the simplest way to divide the image into two clusters would be to take an intensity threshold value and to consider all points having a higher intensity as one cluster and the remaining points as the second cluster. However, the choice of this threshold value can be challenging since there is no obvious way to know exactely the value which would provide the best separation between both clusters. The result of using an appropriate threshold value is illustrated

Figure 2: Example of an input image

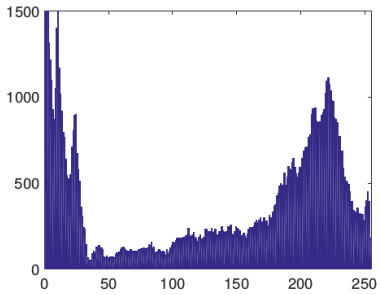

Figure 3: Intensities histogram

Figure 4: Clustering result

in fig. 4. The purpose of this lecture is to introduce an algorithm which would define this threshold value (or a separatrix in the general multi-dimensional case) between the different clusters. This algorithm is called k-means and will be explained in the next section.

4 K-Means Algorithm

This section introduces the intuition and the optimization approach behind the main algorithm presented in this lecture and some of its properties.

4.1 Intuition

We start this section by explaining the intuition behind the K-means algorithm and going through a two-dimensional clustering example using this algorithm.

Algorithm 1 K-Means/Lloyd's Algorithm

1: **function** K-MEANS(K)
2: **Initialize**
Choose K random points to be the initial cluster centers μ_k
3: **repeat**
4: Assign data points $x^{(i)}$ to closest cluster center according to some metric
5: Update each cluster center to be the average of all its assigned points
6: **until** convergence

As discussed in the last section, the final purpose of the clustering process is to assign each data point $x^{(i)}$ of the input dataset to one cluster of K different clusters (where the number of clusters K is given) [4]. The approach of the K-means algorithm is to identify each cluster by a cluster centroid μ_k. Therefore, the first step is to choose randomly K points in the input space to act as the initial cluster centers. The initialization step can have an impact on the final clustering result and will be discussed more precisely in the section 5.1.
Given the initial centroids, the iteration step begins by assigning each input data point $x^{(i)}$ to the closest cluster center using some distance metric, e.g. the Euclidean distance in the input space. In this case, the chosen cluster center for the data point $x^{(i)}$ is the result of computing

$$\arg\min_k ||x^{(i)} - \mu_k||_2^2 \tag{1}$$

Once every data point was assigned to the closest centroid, the cluster centers can be updated by computing the average of all points assigned to it

$$\mu_k = \frac{1}{N_k} \sum_{x^{(i)} \, in \, cluster \, k} x^{(i)} \tag{2}$$

This process of iterating both steps eventually converges when all points' assignments do not change anymore. The introduced algorithm is explained using a simple two-dimensional clustering problem with 2 clusters in fig. 5.

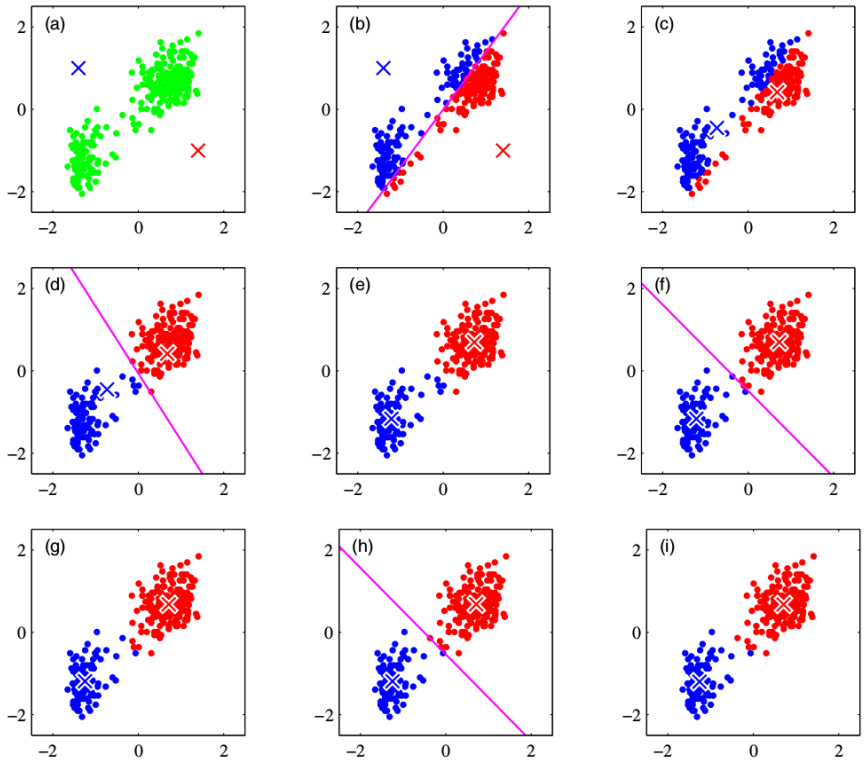

Figure 5: A two-dimensional example of the different steps of clustering using K-means where the number of clusters K is equal to 2: (a) illustrates the initialization step where two random points in the two-dimensional space are chosen to act as the initial cluster centers. (b) The distance between each input data point and both initial cluster centers is computed, and each point is assigned to the closest one (the color indicates the class assignment). (c) Based on the new assignments of points to the clusters, the centroids are updated to be the means of the points of each cluster. (d+e) Second iteration of the algorithm resulting in new cluster centers and in a better clustering. (f+g) Third iteration (h+i) Fourth iteration of the algorithm: note that only one point's assignment has changed compared to last iteration which means the algorithm approaches a local minimum and is very close to converge

4.2 Formal Description

The formal description of the K-means algorithm analyzes mainly the existence of an objective function that the algorithm optimizes.
Before analyzing this objective function, it is useful to introduce a new variable called r_{ik}, where i

denotes the index of the corresponding data point in the input data set and k denotes the cluster. The variable r_{ik} indicates if the data point $x^{(i)}$ is assigned to the cluster centroid μ_k or not. Therefore, it can take only two possible discrete values which are 0 and 1. Additionally, one data point can not be assigned to more than one cluster, which means that the sum of all r_{ik} for any specific data point $x^{(i)}$ over all clusters k should be 1 (In this way and since r_{ik} is either 0 or 1, only one variable among all r_{ik} of some data point $x^{(i)}$ can be one). These constraints related to r_{ik} can be formalized in the following way

$$\begin{cases} r_{ik} \in \{0,1\} & \forall i,k \\ \sum_{k=1}^{K} r_{ik} = 1 & \forall i \end{cases} \tag{3}$$

With this definition of r_{ik}, the cost function of K-means can be formalized to have the following form

$$\min_{\mu} \min_{r} \sum_{i \in \mathcal{D}} \sum_{k=1}^{K} \frac{1}{2} r_{ik} ||x^{(i)} - \mu_k||_2^2 \quad \text{s.t.} \quad \begin{cases} r_{ik} \in \{0,1\} & \forall i,k \\ \sum_{k=1}^{K} r_{ik} = 1 & \forall i \end{cases} \tag{4}$$

The meaning of this cost function, the optimization approach and the reason why it results in the K-means algorithm will be explained in the following lines.
The first observation is that two minimizations are needed unlike most of the optimization objectives we have been dealing with in the first part of this class. An analytical solution for this problem is most likely hard to find, however it is possible to use an optimization technique which deals with each minimization individually and results in a closed-form solution for each minimization objective. This technique is called **alternating optimization**. It consists of applying two consecutive optimization steps at each iteration:

- **Optimize with respect to r given μ**: In this part, the cluster centroids are taken as given, so that the minimization concerns only the variables r_{ik}, i.e. the assignments of each data point to the clusters. Since the r_{ik} of two data points are completely independent of each other (i.e. the assignment of a data point to a cluster does not have an impact on the assignment of another data point to a cluster if the centroids μ_k are given), this problem is equivalent to minimizing the following expression

$$\min_{r} \sum_{k=1}^{K} r_{ik} ||x^{(i)} - \mu_k||_2^2 \quad \text{s.t.} \quad \begin{cases} r_{ik} \in \{0,1\} & \forall i,k \\ \sum_{k=1}^{K} r_{ik} = 1 & \forall i \end{cases} \tag{5}$$

  for each data point $x^{(i)}$. This minimization objective can be achieved by **inspection**. For the i-th point, the constraints indicate that only one of all the variables r_{ik} can be non-zero. We denote with r_{ik^*} this single non-zero item. Therefore, this sum of distances in eq. 5 consists only of one distance corresponding to the $r_{ik^*} = 1$ and can be therefore written as $||x^{(i)} - \mu_{k^*}||_2^2$. This means that the optimization task consists of choosing the cluster k^* for which $r_{ik^*} = 1$. Since there are K possibilities (clusters), all K distances between the data point and the k-th centroid should be computed, and the minimal distance should be chosen for k^*. Any other choice of k^* will result automatically in a greater cost function value. All variables r_{ik} for the i-th point and for $k \neq k^*$ are then set to zero, and the solution which minimizes the eq. 5 for given cluster centroids has the following form:

$$r_{ik} = \begin{cases} 1 & if \quad k = \arg\min_{k \in 1,\ldots,K} ||x^{(i)} - \mu_k||_2^2 \\ 0 & otherwise \end{cases} \tag{6}$$

  Therefore, this step is identical to the first step (fourth line in the pseudo-code 1)in each iteration of the K-means algorithm.

- **Optimize with respect to μ given r**: In this part, it is possible to calculate the derivative with respect to μ and set it to zero, the obtained result has the following form (proof follows):

$$\mu_k = \frac{\sum_{i \in \mathcal{D}} r_{ik} x^{(i)}}{\sum_{i \in \mathcal{D}} r_{ik}} \tag{7}$$

This equation is actually identical to the second step (fifth line in the pseudo-code 1) in each iteration of the K-means algorithm. The nominator of the right side is the sum of all data points assigned to the cluster k whereas the denominator is just their number. As a result, the equation just updates μ_k to be the mean of all points assigned to the cluster k.
Proof: The purpose is to prove that minimizing eq. 4 with respect to μ_k (when r is given) results in eq. 7. First, we calculate the derivative:

$$\frac{\partial}{\partial \mu_k} \sum_{i \in \mathcal{D}} \sum_{m=1}^{K} \frac{1}{2} r_{im} ||x^{(i)} - \mu_m||_2^2 = \sum_{i \in \mathcal{D}} \frac{1}{2} r_{ik} \frac{\partial}{\partial \mu_k} ||x^{(i)} - \mu_k||_2^2 \tag{8}$$

$$= \sum_{i \in \mathcal{D}} r_{ik} (\mu_k - x^{(i)}) \tag{9}$$

In the first line, we removed all items of the sum which do not depend on μ_k (derivative is zero) and only one item is left (the one corresponding to μ_k). In the second line, we applied the rule for the derivative of a squared function. To get the μ_k which minimizes the objective function, the derivative is set to zero and the following equation is obtained:

$$\sum_{i \in \mathcal{D}} r_{ik} \mu_k = \sum_{i \in \mathcal{D}} r_{ik} x^{(i)} \tag{10}$$

Since μ_k does not depend on i, this equation could be written in the following way:

$$\mu_k \sum_{i \in \mathcal{D}} r_{ik} = \sum_{i \in \mathcal{D}} r_{ik} x^{(i)} \tag{11}$$

Dividing by $\sum_{i \in \mathcal{D}} r_{ik}$ on both sides results in eq. 7.

4.3 Properties

This subsection provides an overview of some properties of the K-means algorithm.

- The first property can be concluded almost straight forward from the algorithm and concerns the **running time or complexity**. We discussed in the last subsections that each training iteration consists of two steps. The first step assigns the data points to the closest cluster center. The complexity of this step is $\mathrm{O}(KNd)$ since we have to iterate over all N input data points, and then over all K clusters for each data point to find the cluster center with the minimal distance to each point. The computation of each of the KN distances requires a complexity of $\mathrm{O}(d)$ where d is the dimension of the input space (i.e. of each data point). The second step is just the computation of the average of the data points assigned to each cluster, therefore we will need to iterate once over all input data points which results in a complexity of $\mathrm{O}(N)$.
- The K-means algorithm is guaranteed to converge to a **local minimum and in a finite number of iterations**. In fact, the value of the cost function either decreases or stays the same at each step (see the previous subsection where both minimization steps in the alternating optimization algorithm can never result in a higher cost value), and it is obviously bounded below. However, it is possible that it does not converge to the global minimum.

5 K-Means Extensions

Many attempts were performed to develop extensions and optimized versions of the K-means algorithm. In the following, some of these extensions will be discussed.

5.1 K-Means++

The K-means algorithm does not guarantee the convergence to a global minimum, however just to a local minimum. This makes it very sensitive to initialization because for different initializations the algorithm can converge to different local minima. Therefore, a good initialization is one of the most important challenges. Arthur et al. [1] proposed an optimized version of K-means called K-means++ where the initialization step is performed in a careful manner. The algorithm 2 introduces the K-means++ algorithm where $D(x)$ denotes the shortest distance from a data point to the closest center we have already chosen.

Algorithm 2 K-Means++

1: **function** K-MEANS++(K)
2: **Initialize**
Choose uniformly at random the first center μ_1
3: **repeat**
4: Choose a new center μ_k choosing x from the input space $\mathcal{X}$ with probability $\frac{D(x)^2}{\sum_{x \in \mathcal{X}} D(x)^2}$
5: **until** K centers are chosen
6: Proceed as with the standard K-means algorithm

The use of the K-means++ algorithm leads to an $O(logK)$ approximation of the optimum. In fact, the K-means++ initialization makes it possible to have an upper bound to the expected value of the loss function at convergence. This upper bound is defined to be $8(\ln K + 2)$ of the loss function of the optimal clustering (Proof in [1]). An improved parallelized scalable version of the K-means++ algorithm was introduced in [2].

5.2 Other Distance Metrics and Kernels

The K-means algorithm depends on the use of a distance metric $d(x^{(i)}, \mu_k)$ which measures how a data point and a cluster center are similar or dissimilar with each other. The most commonly used distance measure is the Euclidean distance as introduced in the last sections. However, it is possible to use other distance metrics which can deliver better results dependently on the considered clustering problem. Loohach et al. [3] discusses the effect of using different distance functions (Manhattan distance) on the clustering results of the K-means algorithm. Additionally, it is possible to use the so-called kernel trick by implicitly mapping the input data points to a another feature space (potentially infinite in dimension) and apply the standard algorithm there, i.e. a function ϕ maps each data point $x^{(i)}$ to $\phi(x^{(i)})$ in the new feature space. The K-means algorithm is applied to the points $\{\phi(x^{(i)})\}$. However, the explicit computation of $\phi(x^{(i)})$ is not required (in many cases not possible especially if the feature space has an infinite dimensionality) as long as the kernel $k(x, y) =< \phi(x), \phi(y) >$ can be computed in the new feature space (because this is sufficient to compute the distance).

5.3 Evaluation

The evaluation of the clustering results depends on the context of the application of the algorithm. We distinguish two cases:

- Generative: In this case, we are interested in clustering the data for the sake of regenerating similar data from the obtained centroids. This is also referred to as vector quantization. Each data point $x^{(i)}$ is encoded to the closest centroid μ_{z_i} (z_i denotes the index of the closest centroid to the point $x^{(i)}$). The quality of the clustering result can be evaluated by the reconstruction error (distortion) given by

$$\frac{1}{N}\sum_{i=1}^{N}||x^{(i)}-\mu_{z_i}||^2 \tag{12}$$

- Discriminative: In this case, the true classes are given. Therefore, the quality is measured by labels purity. In fact, each resulting cluster after convergence is assigned to the class which is most frequent in this cluster (e.g. if a cluster contains three data points of class 0 and one data point of class 1 then it is assigned to class 0). The purity measures the accuracy of this assignment by counting the number of correctly assigned data points and dividing by the total number of data points N.

6 K-Means Applications

Two applications of the K-means algorithm are presented in the following lines.

Image Segmentation In the first sections of this lecture, the introduction of the K-Means algorithm was motivated by the aim of clustering the different pixels of an image to multiple regions in order to identify different objects for instance. In fact, image segmentation is one of the most popular application fields of K-means [5].
The following figures illustrate an image segmentation example, where we consider two clustering tasks using the same input image. On the one hand, only the intensities of the pixels are considered which results in a one-dimensional input for each pixel ($x^{(i)} \in \mathbb{R}$). The three resulting clusters are illustrated in fig. 7. On the other hand, all three channels (colors) of the pixels are considered which results in a three-dimensional input for each pixel ($x^{(i)} \in \mathbb{R}^3$). The three resulting clusters are illustrated in fig. 8. In both cases, the spatial smoothness is a problem (i.e. too many outliers for each cluster).

Figure 6: Input image

Figure 7: Clustering (1d grayscale input)

Figure 8: Clustering (3d color input)

Superpixel Segmentation In order to have a better spatial smoothness, the feature space is augmented to contain the spatial coordinates (x, y) of the pixel in the image in addition to the three intensity values [6]. However, the lab color dimensions and the spatial dimensions are treated differently by the distance metric in order to provide a realistic behavior.The results of applying superpixel segmentation on a test image are illustrated in fig. 9.

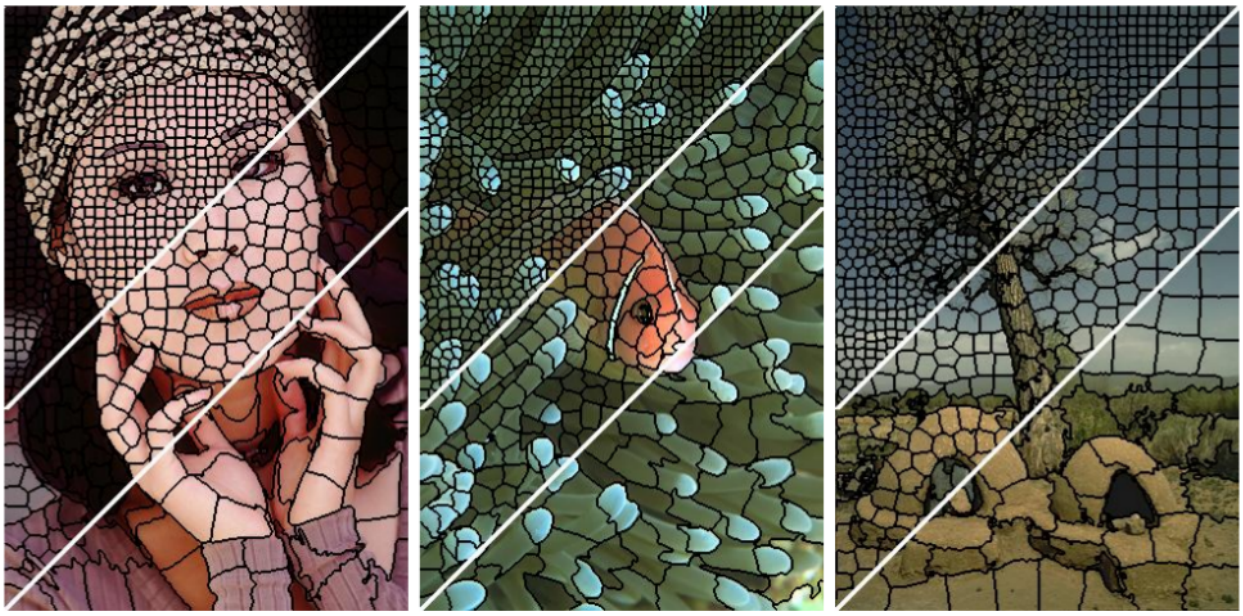

Figure 9: Superpixel Segmentation

7 Conclusion

We conclude with a brief summary of the advantages and downsides of the K-means algorithm. One of the most important aspects of K-means is its simplicity. In fact, it can be implemented very easily in a few lines of code. However, it is sensitive to outliers and to initialization and can get stuck in local minima since there is no guarantee that the global minimum can be found. Additionally, the number of clusters K is taken as input by the algorithm and have therefore to be chosen manually. The non-adaptivity of K-means (same parameters for all clusters) can lead to unrealistic results.
The K-means algorithm makes hard assignments of each data point to some cluster center (r_{ik} is either 0 or 1). It is possible to change these hard assignments to soft assignments so that we obtain a probabilistic model. This will be discussed in the next lecture under the topic of Gaussian Mixture Models.

References

[1] D. Arthur and S. Vassilvitskii. k-means++: The advantages of careful seeding. In *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 1027–1035. Society for Industrial and Applied Mathematics, 2007.

[2] B. Bahmani, B. Moseley, A. Vattani, R. Kumar, and S. Vassilvitskii. Scalable k-means++. *Proceedings of the VLDB Endowment*, 5(7):622–633, 2012.

[3] R. Loohach and K. Garg. Effect of distance functions on k-means clustering algorithm. *Int. J. Comput. Appl*, 49(6):7–9, 2012.

[4] K. P. Murphy. *Machine Learning: A Probabilistic Perspective*. The MIT Press, 2012.

[5] H. Ng, S. Ong, K. Foong, P. Goh, and W. Nowinski. Medical image segmentation using k-means clustering and improved watershed algorithm. In *Image Analysis and Interpretation, 2006 IEEE Southwest Symposium on*, pages 61–65. IEEE, 2006.

[6] S.-C. Wei and T.-J. Yen. Superpixels generating from the pixel-based k-means clustering. *JMPT*, 6:77–86, 2015.