Discovery of Number of Clusters for an Unsupervised Learning

Abstract. Clusters can be found using K-Means Clustering Algorithm. This algorithm requires the input data set and the number of clusters in the dataset. In this paper, we study the behavior of K-Means algorithm with respect to number and choice of the initial cluster centers. We present an algorithm to discover the number of clusters in the given data. We apply this algorithm along with K-Means to demonstrate an entirely unsupervised learning.

Keywords: Dataset, K-Means Clustering Algorithm, Cluster, Cluster Center, Similarity Metric, Euclidean-Distance Metric, Unsupervised Learning.

1 Introduction

Clustering is the process of grouping a set of physical or abstract objects into a set of similar objects. A cluster is a collection of data objects that are similar to one another and are dissimilar to the objects present in other clusters **[1]**. Similarity metrics are used to measure the similarity of the objects and decide which objects are to be clustered. Clustering makes the task of knowledge collection simpler in cases that have massive amounts of data to be analyzed. Many clustering algorithms such as K-Means algorithm, Density-Based clustering methods, K-Medoids method, Hierarchical clustering were studied, where each method is used for a particular purpose.

1.1 The K-Means Clustering Algorithm

In K-means clustering, a set of n data points in d-dimensional space R_d, and an integer k are given, where the problem is to determine a set of k points in R_d, called centers, to minimize the mean squared distance from each data point to its nearest center. A popular heuristic for k-means clustering is Lloyd's (1982) algorithm **[2]**. A candid and efficient implementation of Lloyd's k-means clustering algorithm, called the filtering algorithm is stated in **[3]**. *K*-means clustering aims to partition the *n* observations into k ($\leq n$) sets $\mathbf{S} = \{S_1, S_2, \ldots, S_k\}$ to minimize the within-cluster sum of squares, this is also equivalent to maximizing the squared deviations between points in different clusters. The problem is computationally difficult (NP-hard)**[4]**; however, there are efficient heuristic algorithms that are employed and converge quickly to a local optimum. These are usually similar to the expectation-maximization algorithm for mixtures of Gaussian distributions via an iterative refinement approach employed by both algorithms. Additionally, they both use cluster centers to model the data; however, *k*-means clustering tends to find clusters of comparable spatial extent, while the expectation-maximization allows clusters to have different shapes.

1.2 Similarity Metrics

The similarity of objects is defined in terms of how ‘close' the objects are in space based on the distance function **[5]**. However, various other methods for computation of similarity exist. Distance-Based Similarity Measure is one of the oldest, and the conventional idea of defining similarity, based on the distance between the two object in the space. Distance-based similarity measure [5] is calculated as the Euclidean distance between the two points in the space. Mahalanobis distance is a distance measure, where a point P present in a distribution D is computed as the number of standard deviations that the point is from the mean of the distribution (D) **[6]**. Feature-Based Similarity Measure **[7]** calculates the likeness between two video time series based on the number of common features between the two series considering the objects and their relative spatial and temporal positions extracting spatio-temporal invariant features. Probabilistic Similarity Measure **[8]** use category-specific parameters. These parameters are first and second order statistical parameters based on feature distributions of predefined categories on Gaussian multivariate data.

2 Related Works

When the number of clusters in the given data is not known, various authors **[9-11]** have proposed Elbow method, Average Silhouette method, and X-Means Clustering respectively. In the elbow method [9], the ideal number of clusters is identified by computing the clusters for different values of k(number of clusters) and then calculating the total within-cluster sum of squares(**WSS**). Then we plot a graph with the number of clusters on the X-axis and corresponding WSS on the Y-axis to find the "elbow" shaped bend, which is considered as the optimal number of clusters. In average silhouette method [10], the average silhouette criterion is used to identify the optimal number of clusters. The silhouette of a given data set is the measure of how closely or loosely related are the elements present in it. X-Means Clustering [11] is similar to K-Means clustering except that it takes into account the ideal number of clusters, which is identified by iteratively attempting sub-division among the clusters to find the optimal number of clusters, which satisfies the Akaike information criterion or the Bayesian information criterion.

3 Motivation

Many methods were proposed to determine the number of clusters and initial cluster centers. Some methods to define the number of clusters are initialization method, encoding method, and tentative clustering. Methods for initial cluster center selection are Refining Initial Points Algorithm **[12],** Cluster Centroid Decision Method**,** Cluster Seed Selection **[13].** According to these methods, the K-Means algorithm would perform better when it has a routine that is capable of defining the number of clusters ideal for the given data set, and a routine that would determine the ideal initial cluster centers. These works have motivated us to focus on developing an unsupervised learning to discover the number of clusters required to run K-Means.

Problem Statement:
Discover the number of clusters in given data and find suitable initial cluster centers to run K-

Means algorithm, and thereafter to establish the members of each cluster.

Our Contribution:

1. Developed an algorithm which can be used to cluster data points without prior knowledge on the number of clusters.

2. This algorithm enables discovery of relevant cluster sets whose final utility can be determined by the domain expert.

4 An Illustration to understand working of K-Means Clustering Algorithm

K-Means algorithm:

K-Means algorithm is a contribution by Lloyd **[3]**.

```
Algorithm K-Means Algorithm
Input:
    • k: the number of clusters
    • a data set D containing n objects.
Output: A set of k clusters
Procedure:
    1. Arbitrarily choose k objects from D as initial cluster centers
    2. Repeat
    • Assign each object to the cluster to which the object is the most similar, based on
      the mean values of the object in the cluster
    • Update the cluster means, i.e., calculate the mean value of the objects for each
      cluster
    3. Until no change
```

Fig 1. K-Means clustering algorithm

Consider an example case that implements K-Means algorithm to perform clustering of the data set: (0,0), (2,2), (4,6), (3,5), (5,6), (3,4), (5,2), (4,1), (1,1), (5,0), (6,0), (0,2). We have chosen the number of clusters as 3 and initial cluster centers as (0, 0), (3, 5), (5, 0). In the below tables, K_j^n denotes the cluster center of cluster j for the iteration n and $\left|p_i - K_j^n\right|$ denotes the Euclidean distance of data point p_i from the cluster center K_j^n. The results are shown in Figure 2 below. C^i_j represents the cluster 'j' formed after 'i' iterations.

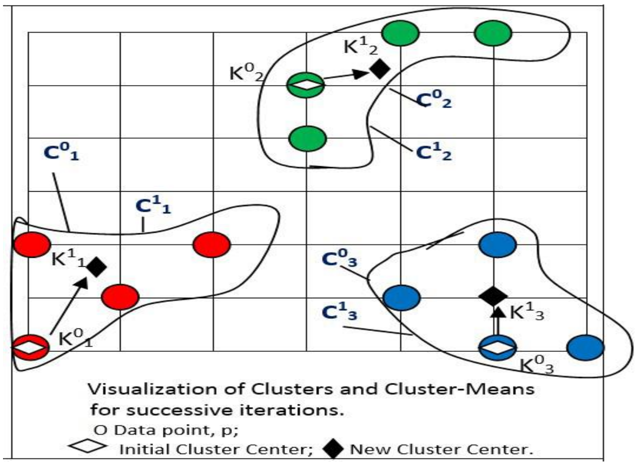

Fig 2. Cluster elements and cluster centers for given data points

5 Requirements for K-Means Clustering Algorithm

K-Means Clustering Algorithm only works when the number of clusters and the initial cluster centers is declared before the commencement of the algorithm. We present a study of how the choice of centers affects the results of K-Means Clustering Algorithm. The below-given table shows the various cases that were studied when the algorithm was successful or unsuccessful in clustering the data. In the below table, we can observe that the algorithm does not work if two or more cluster centers are chosen to be the same data point. In our study, we have considered the number of clusters as 2,3 and 4. If we choose the number of clusters to be 3, and two of three cluster centers are chosen from the same cluster, then the algorithm is successful in generating the clusters indicated as S in Table 1. Similarly, if we choose the number of clusters to be 4, and 3 of 4 cluster centers are chosen from one cluster, then the algorithm is successful. However, if we choose the number of clusters as 4 and if two cluster centers were the same, then the algorithm fails indicated by F in Table 1.

Table 1. Results of test cases to understand the k-means algorithm

Number of Clusters	One from each cluster	Different centers			Same centers		
		Two from each cluster	Three from each cluster	Four from each cluster	Two from each cluster	Three from each cluster	Four from each cluster
2	S	S	-	-	F	-	-
3	S	S	S	-	F	F	-
4	S	S	S	S	F	F	F

6 Discovery of Number of Clusters

As discussed in the preceding sections, the K-Means algorithm performs clustering of data set when the number of clusters and initial cluster centers is declared. It does not provide any insight into how clustering can be done for cases where the number of clusters is not declared. Our aim is to discover the number of clusters in given data and find suitable initial cluster centers to run K-Means algorithm and establish the members of each cluster. We present a few novel ideas that are useful in the development of the algorithm that we want to develop.

6.1 Connectivity Test:

A set of points is said to be connected if a point has at least one more point at a distance less than or equal to a predefined threshold distance. The distance between adjacent pairs of points of a given cluster should be less than or equal to a threshold distance.

Determining the threshold distance: The connectivity test was formulated based on the ideology of constructing a unit square around all the given data points. Now all the data points whose squares are in contact by either their sides or their corners are considered to be connected. If the length of each side of the square was of 1 unit length, then the maximum distance between any two points which are connected is $\sqrt{2}$. Therefore, the threshold distance is $\sqrt{2}$. However, the length of the side of the square cannot be considered as 1 unit length for all the cases. This is because, if there was a dataset, which has data points spread in such a way that no two points are at a distance of 1 unit length, then no two data points will exist in the same cluster according to the connectivity test. Hence, the length of the side of the square varies for each data set. To accommodate this, the algorithm initially assigns $\sqrt{2}$ as threshold distance, and for every iteration where the connectivity test fails, the threshold distance is incremented by $\sqrt{2}$. Finally, the algorithm applies connectivity test on the clusters with the new threshold distance to check whether they are connected. This process is repeated until the connectivity test passes, or the threshold distance(d) is greater than the upper threshold distance(q), where q is the distance between the two farthest points present in the dataset.

```
Algorithm: Connectivity Test
Input: Set of clusters (C), Threshold Distance (δ).
Output: Connectivity of the clusters.
Procedure:
    1. For every cluster $C_i$ present in C, calculate the Euclidean distance between the points
       in the cluster to check if it is less than or equal to threshold distance δ.
    2. If all clusters have data points within the distance threshold, then return true.
    3. Else return false
```

Fig 3. Algorithm for connectivity test.

Consider the test data plotted in Figure 4 below, the connectivity test for a threshold distance equal to $\sqrt{2}$ for the points (3, 4) and (3, 5). This pair of points is connected as they are within $\sqrt{2}$ distance. Similarly, this test on the points (0, 0), (0, 1), (1, 1) and (1, 2) yield a result as connected.

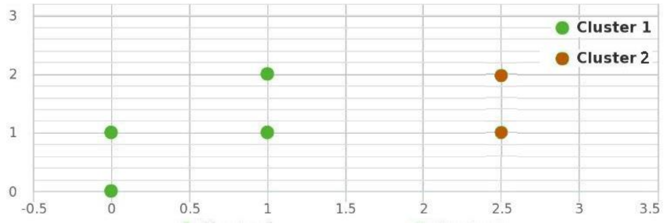

Fig 4. An example case for a cluster that does not satisfy the connectivity test

6.2 Stop Condition:

Stop condition determines when the algorithm terminates. If clusters satisfy the connectivity test and all data points are allotted to clusters, the algorithm terminates. If the clusters do not satisfy the connectivity test, then increase the threshold distance and re-check if the clusters pass the connectivity test. This process is repeated until all clusters satisfy the connectivity test or the threshold distance becomes greater than an upper threshold (q) this is when finally the algorithm terminates. Using these concepts, we present an algorithm that discovers the number of clusters as shown in Figure 5.

Proposed Algorithm:

```
Algorithm:  Number of clusters discovery using K-Means
Input: Dataset D of n data points each having d dimensions.
Output: Set of number of clusters, cluster members and threshold distance (δ)
Procedure:
    1.  Assume the initial number of clusters N as 2, where $2 \leq N \leq n$.
    2.  Assign the threshold distance (δ) to $\sqrt{d}$
    3.  Generate N cluster centers randomly in the data set space.
    4.  Run the K-Means clustering algorithm with these centers.
    5.  Apply connectivity test for members of each cluster using a threshold distance
        value, δ [p, q].
    6.  If the test passes, go to step 9.
    7.  Else, increment the threshold distance(δ) by $\sqrt{d}$ and
        repeat
              step 5
        until
              the connectivity test passes
            or
              the value of distance metric is equal to an upper threshold value (q).
    8.  If connectivity test passes, then go to step 9.
        Else,
            If N<n, then increment N by 1 and go to step 3.
            Else if $N \geq n$, then go to step 10.
    9.  Print the number of clusters, cluster members, and threshold.
    10. Print that the algorithm could not find the clusters.
```

Fig 5. Number of cluster discovery using K-Means

6.3 Number of Clusters vs Upper Threshold Distance:

Apply the algorithm to the dataset and note down the number of clusters formed for each corresponding upper threshold distance (q). Finally, plot a graph with Number of Clusters (N) on the X-axis and Upper Threshold Distance (q) on Y-axis. The plot shows a graph where the number of clusters decreases as upper threshold distance increases. For illustration, consider the Fisher iris dataset which contains 150 data points and yields a graph as below, when given as input to the algorithm. From the below graph, it can be derived that 2 and 3 are the number of clusters that can be generated from the given dataset. However the minimum number of clusters that can be generated is 2.Ttherefore, the ideal number of clusters for the given dataset is 3.This case is used as our case 1 in the next section.

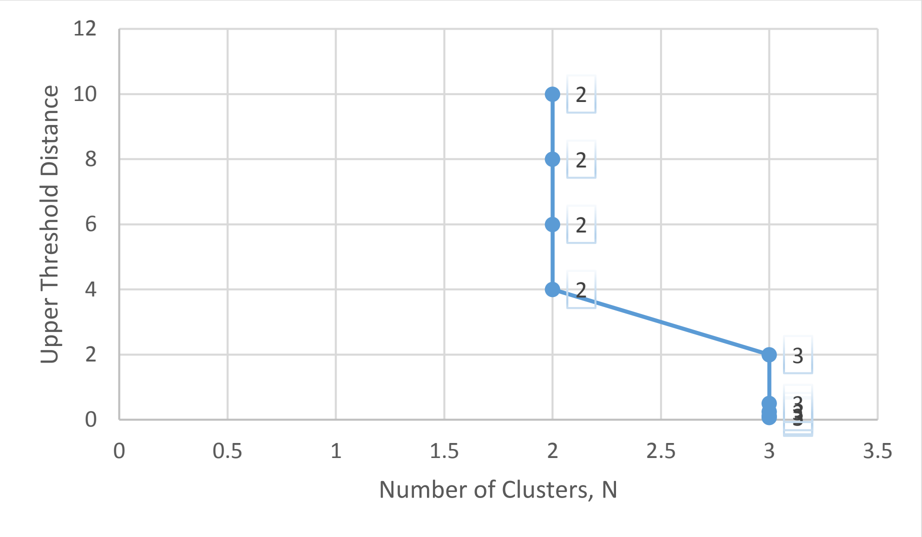

Fig 6. Number of Clusters vs Upper Threshold Distance graph for Fisher Iris Dataset

7 A Study to Illustrate the Working of the Proposed Discovery Algorithm

This study includes data sets drawn from the UCI Repository. The first data set is Fisher Iris dataset [14], the second data set is the glass identification data set [14] and the third data set is seeds data set. The Number of Clusters (N) vs Upper Threshold distance (q) graphs for the dataset is shown in Figures 7, 8, 9 for cases 1, 2 and 3 respectively. The algorithm was successful in presenting the minimum number of clusters required using threshold value as $\sqrt{d}$, where d is the number of dimensions present in each data points.

Case 1. Fisher Iris: The fisher iris dataset consists of 150 data points each consisting of four dimensions. This dataset consists data points corresponding to three different types of iris plants. The algorithm generates a graph as shown in figure 6, which shows that the ideal number of clusters for this data set is three.

Case 2. Glass Identification Dataset

The glass identification dataset consists of 214 data points where each data point has nine dimensions. The dataset consists of six different types of glass according to [14].

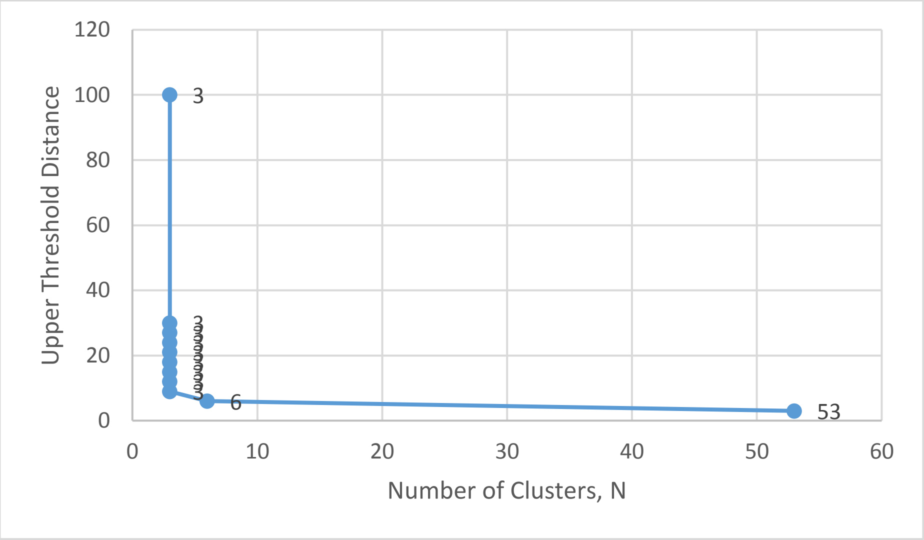

Fig 7. Number of Clusters vs Upper Threshold Distance for Glass Identification dataset

Case 3. Seeds Dataset

The seeds dataset consists of 210 data points where each data point has seven dimensions. This data set provides measurements of geometrical properties of three different varieties of wheat. The below graph shows that the number of clusters ideal for this data set is three as stated by the source of this dataset.

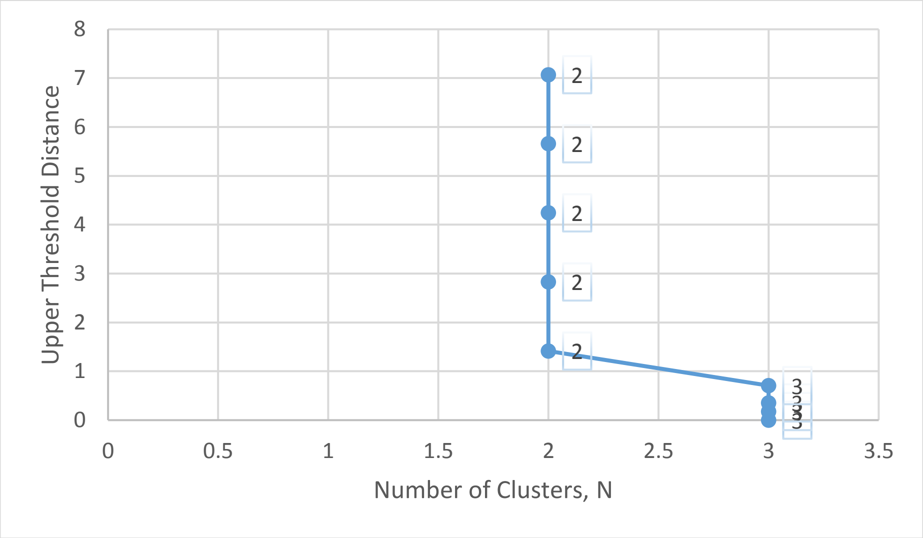

Fig 8. Number of Clusters vs Upper Threshold Distance for Seeds dataset

Case 4. Haberman's Survival Dataset

The survival dataset consists of 306 data points where each data point has three dimensions. This dataset provides cases pertaining to the survival of patients suffering from breast cancer. The below graph shows that the number of clusters ideal for this data set is two as stated by the source of this dataset.

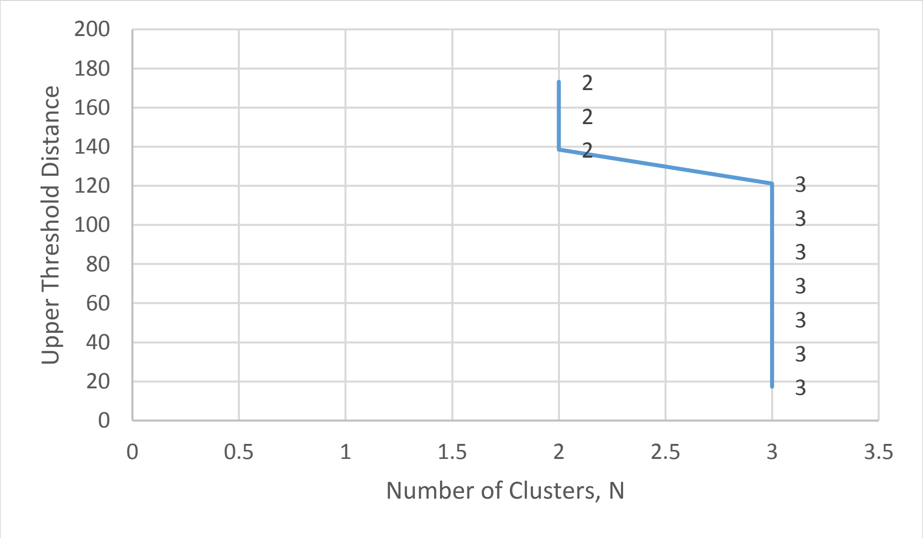

Fig 9. Number of Clusters vs Upper Threshold Distance for Haberman's Survival data set

From the cases 1 to 4 the promising number of clusters are 2,3 and 6. However, hypothetically, this could have been any number of clusters. The expert user now has the ability to focus on these promising clusters i.e. 2 or 3 or 6 as the case may be. Among these clusters, the experts would use a set of clusters that suit their requirement best.

8 Conclusion and Future Works

We attempted to develop an algorithm that discovers the number of clusters in given data set. Our attempt was to use K-Means algorithm along with the algorithm we proposed. The first one establishes the number of clusters and the second establishes members in each cluster. Together we get an automated version that does not require any supervision to find clusters. Thus, this work improves the use of K-Means and enables automatic cluster discovery given the data set. We would like to extend this work that attends to compare our method with other methods and also study the effect of scaling the algorithm.

9 References

1. J. Han, M. Kamber, "Data Mining – Concepts and Techniques", *Morgan Kaufmann Publishers*, 2006.
2. Lloyd, Stuart P., "Least squares quantization in PCM", *IEEE Transactions on Information Theory,* 1982.
3. T. Kanungo, D.M. Mount, N.S. Netanyahu, "An efficient k-means clustering algorithm: analysis and implementation", *IEEE Transactions on Pattern Analysis and Machine Intelligence, Jul 2002.*
4. Dasgupta. S, "The hardness of k-means clustering", *Technical Report, University of California, 2008.*
5. F. Gregory Ashby; Daniel M. Ennis, "Similarity measures", *www.scholarpedia..org/article/Similarity_measures#A_Classification_of_Similarity_ Modelscom,* 2007.
6. P. C. Mahalanobis, "On the generalized distance in statistics", *Proceedings of the National Institute of Sciences of India*, 1963.
7. Sengupta S., Wang H., Blackburn W., Ojha P., "Feature-Based Similarity Measure", Conference Paper, *Springer-International Conference on Ubiquitous Computing and Ambient Intelligence, 2014.*
8. Rahman M.M., Bhattacharya P., Desai B.C., "Probabilistic Similarity Measures in Image Databases with SVM Based Categorization and Relevance Feedback", *Conference Paper, Springer-International Conference on Image Analysis and Recognition,* 2005.
9. Robert L. Thorndike, "Who Belongs in the Family?". *Psychometrika, Springer,* 1953.
10. Peter J. Rousseuw, "Silhouettes: a Graphical Aid to the Interpretation and Validation of Cluster Analysis". *Computational and Applied Mathematics – Springer, 1987.*
11. Pelleg; AW Moore. X-means: Extending K-means with Efficient Estimation of the Number of Clusters, *Proceedings of the Seventeenth International Conference on Machine Learning (ICML), 2000.*

12. P.S. Bradley; Usama M. Fayyad, “Refining Initial Points for K-Means Clustering”, *ACM Digital Library, 1998.*

13. Fouad Khan, “An Initial Seed Selection Algorithm for K-means Clustering of Georeferenced Data to Improve Replicability of Cluster Assignments for Mapping Application”, *Cornell University Library, 2016.*

14. Lichman, M. (2013). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.