Project 2: Structure in compressed spaces (Unsupervised Learning). Assignment 1.

Student number: 1802493

***Abstract*—Clustering is one of the most common challenges in machine learning. To find useful patterns in unlabeled data is not an easy task. Many techniques have been developed for this purpose over the last century. This project uses a state-of-art method for clustering: the traditional algorithm k-means complemented with autoencoders. This allows us to deal with the problems related to high-dimensional and unstructured data. Three datasets of very different nature were chosen to test this method. The first one is a database of images of handwritten numbers (PenDigits). The second is a dataset of sensor outputs registered while performing different human activities (HHAR). The third is a database of mushrooms types with their descriptive features (Mushrooms). To assess the performance of the proposed method, an evaluation plan that includes internal and external evaluation metrics, is presented.**

I. Introduction

UNSUPERVISED learning can be a difficult task because patterns need to be extracted from the data without having a response variable available for training. Traditional methods for clustering, such as k-means, use distance measures that have some limitations. In some cases, when dealing with image datasets for example (where the features are the raw pixels), using the k-means directly on the original set of features is proven to be ineffective (Zhang et al., 2018 [1]). To deal with this kind of challenges, algorithms may be applied prior to the clustering to determine a new feature space.

This project considers using autoencoders as a feature reduction method to apply in combination with traditional clustering techniques. An autoencoder can be considered as a neural network (NN) that is trained with the original features as both, the input and the output layer, and is composed by encoder/decoder functions in the hidden layers. The encoder generates new latent representation of the data, but with a reduction on dimensionality. By estimating the parameters of the NN jointly with the clusters' centers, the "best" encoder for the purpose of clustering is found.

This first part of the project considers an exploratory analysis of the data and a description and planification of the tasks that will come next. No results of the clustering are presented in this report.

II. Background

Clustering has been the topic of many reasearch studies in the past. Its origins date back to the first half of the 20th century and were driven by researchers in the social sciences, such as anthropology and psychology.[1] Several types of algorithms have been developed since. From k-means (MacQueen et al., 1967 [3]) to DBSCAN (Ester et al., 1996 [4]), many authors have proposed different techniques to find the best clustering method possible. The truth is that all techniques have some advantages and disadvantages and defining which one is best usually depends on the characteristics of each particular problem.

Many of the traditional clustering techniques present problems when dealing with high-dimensionality (Steinbach et al., 2004 [5]) and feature reduction is not a simple task. Principal Component Analysis (Pearson, 1901 [6]) is a traditional approach for feature reduction that transforms the original features into uncorrelated components that are defined in a way that the first components accumulate most of the variance of the data. New approaches propose the use of neural networks, more specifically autoencoders, to deal with the reduction of dimensionality of the datasets. Methods such as Deep Embedded Clustering (DEC) (Xie et al., 2016 [7]), that simultaneously defines the feature reduction and the cluster assignment, have shown significant improvement over other state-of-the-art clustering techniques. This is the approach that will be used in this project.

Some extensions of the DEC technique, such as Variational Deep Embedding (VaDE) (Jiang et al., 2016 [8]) and Gaussian Mixture Variational Autoencoder (GMVAE) (Dilokthanakul et al., 2016 [9]), have been proposed and have also shown good results.

III. Methodology

This section presents the method that will be used for clustering and makes a detailed description of the three datasets that will be used.

A. Method

For this project autoencoders will be used with a traditional clustering algorithm, k-means. A simplified explanation of this technique is presented.

1) Autoencoders: An autoencoder is a type NN where the input is the original list of features (x) and the output is the reconstruction of this list (x'). The intermediate layers are composed by an encoder function and a decoder function. To simplify, we will think about it as a two layers NN, so that $h_1$is the activation function of the encoder and h_2 is the activation function of the decoder. This is represented in the diagram of Figure 1. The encoder is what we will use to reduce the dimensions of the features.

To determine the values of the parameters, a minimization is done over the squared errors. This minimization problem is presented in Equation (1).

[1]The first approach to clustering was done by antropologists Driver and Kroeber in 1932[2].

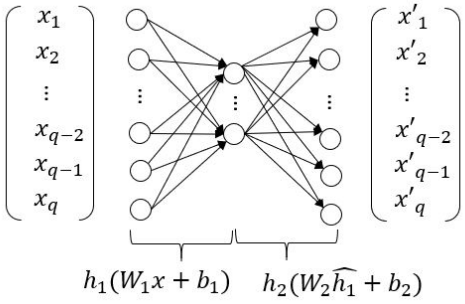

Fig. 1. Autoencoders are NN that have as input the features and as autput a reconstruction of them. To do this , the intermediate layers have an encoder and a decoder. By defining he parameters of the encoder function ($W_1$and b_1) the new lower-dimensionality features can be computed.

$$\min_{W_1,W_2,b_1,b_2} \sum_i (x_i - x'_i)^2 \tag{1}$$

2) K-Means: We use k-means to find the optimal assignment of instances to k clusters. The objective is to find k centroids and assign the instances to the k clusters, so that the sum of the distances of each point to the centroid of the cluster where that point is assigned is minimized. This is presented in Equation (2), where $c_j$is the centroid of cluster $S_j$and x_i is the i-th instance of the cluster S_j.

$$\min_{c_j} \sum_j \sum_{x_i \in S_j} (x_i - c_j)^2 \tag{2}$$

3) Autoencoders+K-Means: When using autoencoders to reduce the dimensionality it is important that both optimization problems (1) and (2) are solved simultaneously. If (1) is solved separately some parameters (W_1, W_2, $b_1$and b_2) will be found, but nothing can assure that those parameters are the ones that will reduce the features' dimensions adequately for our goal, which is to find the clusters.

In order to do the clustering, we need to solve both optimization problems together, as stated in Equation (3).

$$\min_{W_1,W_2,b_1,b_2,c_j} \sum_i (x_i - x'_i)^2 + \sum_j \sum_{x_i \in S_j} (x_i - c_j)^2 \tag{3}$$

B. Datasets

Three very different datasets will be used to evaluate the proposed method. Althought these datasets will be used for unsupervised learning, all of them are labeled. This label will not be used in the training of the models, but will later be used in evaluating the completeness metric. Because of their different nature, the analysis and pre-processing of the data is also different in each case. Table I shows a summary of the final datasets to be used in the cluster analysis. The total number of observations (# of Obs.), Features (# of Feat.) and Classes (# of Class.) is shown. Also, details on these characteristics are presented in the last three rows: the presence of missing values, the type and the scale of the features. These characteristics indicate whether it is necessary to make some pre-processing of the data (like imputations or re-scaling) before using it in the clustering training. A detailed description of every dataset is presented next.

TABLE I
HHAR VARIABLES DESCRIPTION

	PenDigits	**HHAR**	**Mushrooms**
# of Obs.	1,797	4,712	8,124
# of Feat.	64	600	22
# of Clas.	10	6	2
Missing Val.	No	No	Yes
Type of Feat.	Continuous	Continuous	Categorical
Scales of Feat.	Homogen.	Slightly Heterogen.	N/A

1) PenDigits: The original Pen-Based Recognition of Handwritten Digits Data Set contains about 11,000 instances and 16 features. Each instance represents an image of a handwritten object. The instances were collected from 44 different writers who were asked to write 250 digits from 0 to 9. The features are integers that represent coordinate information. Each number is associated with a coordinate (x,y). The initial image has 250,000 (500x500) possible coordinate points, but after processing, the data, each image can be represented as an 8x8 bitmap (64 pixels). The processing of the data is shown in Figure 2.

For this project we will only be using the processed data where every image is represented by 64 pixels. These pixels are the features we will be using in the clustering algorithms. All features in this dataset are continuous and have the same range, as they hold a numerical value from 0 to 16, representing the level of darkness of the grey color in the pixel.

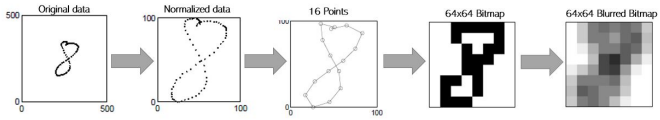

Fig. 2. Processing of original dataset. A normalization is done to reduce dimensions from 500x500 to 100x100. Resampling is applied so that each image can be represented by only 16 strategic points. These points are the transformed into a 64x64 bitmap using Bresenham's Algorithm and then the image is blurred to deal with the small variations on pen's coordinates that could distort the overall image. More detail on how this transformation from points to bitmap is done can be found in (Alimoglu et al., 1996 [10]).

By processing the data set, a simpler representation of the images is obtained. This is convenient to reduce the processing time when training the clustering model. However, the risk of the machine confusing the digits increases. In Figure 3 the plotting of 10 instances of the data set, each representing one digit from zero to nine, is presented. At simple sight, in this example some digits could be easily mistaken. For example, digits 3 and 5 have similar bitmap representations in this case.

For simplicity, in this project a sample of only 1,797 instances from the 11,000 available will be used.[2] A variable

[2]This sample is available as the "digits" data set in the scikit learn library.

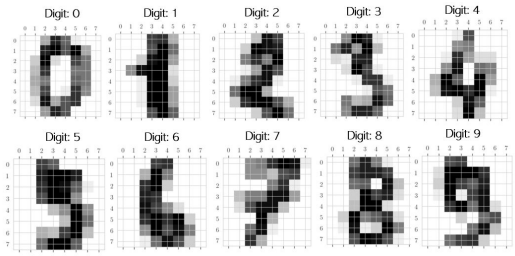

Fig. 3. Plots of a sample of 10 instances of the final database. Due to the low resolution, some digits could easily be confused with one another. In the example, the plots for 3 and 5 look very similar.

description of the final data set can be found in Table II. The distribution of the classes (digits) in the final data set is presented in Table III.

TABLE II
PENDIGITS VARIABLES DESCRIPTION

Variable	Description
pixel_i	Pixel in position i of the 8x8 bitmap. It takes integer values from 0 to 16 which indicate the scale of gray of the pixel (being 0 the lightest and 16 the darkest). i=0,1,..,63.
label	The number which the image represent. It could be any number from 0 to 9.

TABLE III
PENDIGITS CLASS DISTRIBUTION

Class	0	1	2	3	4
Perc.	9.9%	10.1%	9.8%	10.2%	10.1%
Class	5	6	7	8	9
Perc.	10.1%	10.1%	10.0%	9.7%	10.0%

2) HHAR: The Heterogeneity Dataset for Human Activity Recognition contains the sensor outputs from smartphones and smartwatches of people performing different activities. The data set contains the readings of two motion sensors for each type of device: accelerometer and gyroscope. Since the original data set is very large, this project only uses the information of the smartphones' accelerometer. Outputs coming from the smartphones' gyroscope and the smartwatches are dissmissed. The unprocessed data set has 10 variables and 13,062,475 instances. A description of these variables is presented in Table IV.

For simplicity, this project only uses information from device samsungold_1. All instances with activity "null" are also dismissed from the data set.

The data has been restructured into tabular format, with 4,712 instances (each one labeled according to the activity performed) and 600 columns, consisting of the three coordinates x, y, z over a sequence 200 time steps. All features variables in this data set are continuous because they represent coordinates in space. The scales of x, y and z are slightly different from one another. This can be seen in the vertical axis in Figure 4

TABLE IV
HHAR VARIABLES DESCRIPTION

Variable	Description
Index	Row number.
Arrival_time	The time the measurement arrived to the sensing application.
Creation_time	The timestamp the OS attaches to the sample.
x	Value provided by the sensor for coordinate x.
y	Value provided by the sensor for coordinate y.
z	Value provided by the sensor for coordinate z.
User	The user this sample originates from. There are 9 users named a to i.
Model	The phone model the sample originates from. There are 4 models: nexus4, s3, s3mini, samsungold.
Device	The specific device this sample is from. They are prefixed with the model name and then the number, e.g., nexus4_1 or nexus4_2.
gt	The activity the user was performing: bike, sit, stand, walk, stairsup, stairsdown and null.

which takes different ranges of values depending on the type of coordinate.

When observing the sequence of coordinates of the processed data set in Figure 4, one can appreciate different patterns of movement for each activity. For example, the activity "Stand" has very little variation in the coordinates' values over time while the activity "Stairs up" shows ups and downs in regular intervals. By drawing the same plots presented in Figure 4 for different instances, one can see that these patterns seem to repeat themselves when the activity is the same.

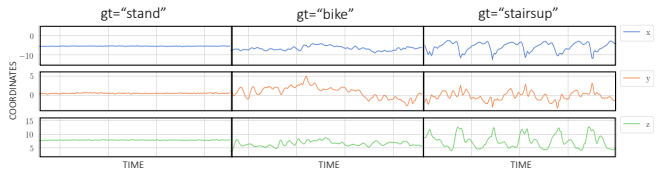

Fig. 4. Sequence of space coordinates by activity. The plot shows 3 samples of instances: one for activity "Stand", one for "Bike" and one for "Stairs up". Each coordinate is plotted over time as a different color line (x=blue, y=orange, z=green). The period of time represented for each activity is 200 sequential time steps.

The distribution of the classes (activities) in the final data set is presented in Table V.

TABLE V
HHAR CLASS DISTRIBUTION

Class	sit	stand	stairsup	stairsdown	walk	bike
Perc.	14.3%	14.7%	16.5%	16.6%	18.8%	19.2%

3) Mushrooms: The Mushroom data set contains the description of samples corresponding to 23 species of mushrooms from the Agaricus and Lepiota Family. It has 8,124 instances and 23 variables (22 features + 1 label). The label

is a binary variable that indicates if the mushroom is edible or not. Since the edibility of the mushrooms can usually be determined by its physical characteristics, we will be using deep clustering to see if these two groups can be identified by only looking at the features. A diagram of the mushrooms' different parts is presented in Figure 5.

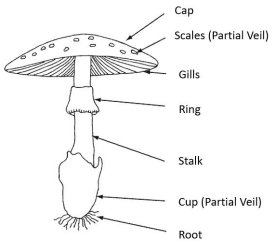

Fig. 5. The parts of the mushrooms that are described by the features of the data set are shown in this diagram.

All the variables of this dataset are categorical.[3] Table VI has a description of these features and their categories.

When analyzing the frequencies of the categories for each feature[4], we can see that for two of the variables more than 95% of the instances are concentrated in only one category: veil_type (100% belongs to category "p") and veil_color (98% belongs to category "w"). All the mushrooms in the data set have partial veil and most of them are white. These variables are of no use for discriminating among clusters and will be removed prior to the training. Also, the feature stalk_shape has 31% of its values missing and it may be necessary to remove or impute it when training the clusters.

The distribution of the classes (edibility) in the final data set is presented in Figure VII.

IV. Experiments

The method proposed (Autoencoders+k-means) will be tested in the three data set previously described. The number of clusters for the k-means algorithm will be defined according to the number of classes of the labels. Since k-means is sensitive to the initial centroids, we will perform 10 random restarts when selecting the centroids and pick the result with the best objective function.

A k-means clustering without autoencoders will also be performed to later assess the effect of using autoencoders. For this task, the same method will be used to select the initial centroids.

The results of these experiments will be presented in following reports.

V. Discussion

For evaluation, two forms of metrics will be used.

First, we will do an internal evaluation of the clusters' assignment using the Silhouette Coefficient (SC). This coefficient contrasts the average distance of instances within the

[3]Only ring number could be considered continuous, but it is coded as categorical an will be used as such.

[4]The distribution of categories for each variable can be found in the Jupyter notebook attached.

TABLE VI
Mushrooms Variables Description

Variable	Description
label	Binary variable that indicates edibility. poisonous=p, edible=e.
cap_shape	Shape of the cap. bell=b, conical=c, convex=x, flat=f, knobbed=k, sunken=s.
cap_surface	Surface of the cap texture. fibrous=f, grooves=g, scaly=y, smooth=s.
cap_color	Color of the cap. brown=n, buff=b, cinnamon=c, gray=g, green=r, pink=p, purple=u, red=e, white=w, yellow=y .
bruises	Binary variable that indicates the presence of bruises. bruises=t, no=f.
odor	Odor of the mushroom. almond=a, anise=l, creosote=c, fishy=y, foul=f, musty=m, none=n, pungent=p, spicy=s.
gill_attachment	Type of attachment of gills. attached=a, descending=d, free=f, notched=n
gill_spacing	Spacing of the gills. close=c, crowded=w, distant=d
gill_size	Size of the gills. broad=b, narrow=n.
gill_color	Color of the gills. black=k, brown=n, buff=b, chocolate=h, gray=g, green=r, orange=o, pink=p, purple=u, red=e, white=w, yellow=y.
stalk_shape	Shape of the stalk. enlarging=e, tapering=t.
stalk_root	Type of root of the stalk. bulbous=b, club=c, cup=u, equal=e, rhizomorphs=z, rooted=r, missing=?.
stalk_surface_above	Stalk surface above the ring. fibrous=f, scaly=y, silky=k, smooth=s.
stalk_surface_below	Stalk surface below the ring. fibrous=f, scaly=y, silky=k, smooth=s.
stalk_color_above	Stalk color above the ring. brown=n, buff=b, cinnamon=c, gray=g, orange=o, pink=p, red=e, white=w, yellow=y.
stalk_color_below	Stalk color below the ring. brown=n, buff=b, cinnamon=c, gray=g, orange=o, pink=p, red=e, white=w, yellow=y
veil_type	Veil type. partial=p, universal=u
veil_color	Veil color. brown=n, orange=o, white=w, yellow=y
ring_number	Number of rings. none=n, one=o, two=t
ring_type	Type of rings. cobwebby=c, evanescent=e, flaring=f, large=l, none=n, pendant=p, sheathing=s, zone=z
spore_print_color	Spore print color. black=k, brown=n, buff=b, chocolate=h, green=r, orange=o, purple=u, white=w, yellow=y
population	Type of population. abundant=a, clustered=c, numerous=n, scattered=s, several=v, solitary=y
habitat	Habitat. grasses=g, leaves=l, meadows=m, paths=p, urban=u, waste=w, woods=d

TABLE VII
MUSHROOMS CLASS DISTRIBUTION

Class	poisonous	edible
Perc.	48.2%	51.8%

same cluster with the average distance to instances in other clusters. Intuition is that clusters with high similarity within a cluster and low similarity between clusters will have higher SC. The SC formula is presented in Equation 4, where a_i is the average distance between point i and all other data within the same cluster and b_i is the smallest average distance of i to all other points in other clusters.

$$s_i = \frac{b_i - a_i}{\max\{a_i, b_i\}} \tag{4}$$

The second form of evaluation will be external, comparing the clusters obtained with the labels. In order to do this, three different metrics will be computed: the Completeness Score (CS), the Homogeneity Score (HS) and the Cluster Accuracy (ACC). The metrics CS and HS are both necessary when evaluating clusters solutions because they run in opposition: Increasing the homogeneity of a clustering solution often results in decreasing its completeness. As explained by (Rosenberg et al., 2007 [11]): "A clustering result satisfies homogeneity if all of its clusters contain only data points which are members of a single class. A clustering result satisfies completeness if all the data points that are members of a given class are elements of the same cluster." The ACC is computed so that the results obtained are comparable to those presented by other authors who used deep clustering on very similar data sets.

$$CS = 1 - \frac{H(K \mid C)}{H(K)} \tag{5}$$

$$HS = 1 - \frac{H(C \mid K)}{H(C)} \tag{6}$$

With:

$$H(C \mid K) = -\sum_{c=1}^{|C|}\sum_{k=1}^{|K|}\frac{n_{c,k}}{n} log(\frac{n_{c,k}}{n_k})$$

$$H(C) = -\sum_{c=1}^{|C|}\frac{n_c}{n} log(\frac{n_c}{n_k})$$

Where $H(C \mid K)$ is the conditional entropy of the classes given the cluster assignments and $H(C)$ is the entropy of the classes, with n the total number of samples, n_c and n_k the number of samples respectively belonging to class c and cluster k, and finally $n_{c,k}$ the number of samples from class c assigned to cluster k.

$$ACC = \max_m \frac{\sum_{i=1}^{n} 1\{l_i = m(c_i)\}}{n} \tag{7}$$

Where l_i is the ground-truth label, c_i is the cluster assignment produced by the algorithm, and m ranges over all possible one-to-one mappings between clusters and labels.

Both, internal and external evaluation are important. On the one hand, internal evaluation will give insights about the construction of the clusters (how cohesive are the clusters and how separate they are from each other) and the external evaluation shows if the clusters are good at predicting what we wanted to predict.

To evaluate the quality of the results, the evaluation metrics will be compared to: 1. The same metrics computed for the clusters obtained when using only k-means, to evaluate the effect of using autoencoders. 2. The ACC metric will be compared to those obtained (Xie et al., 2016 [7]) and (Zhang et al., 2018 [1]), to evaluate how good our method is compared to the ones implemented by other authors.

VI. CONCLUSION

In this first part of the project we have covered the literature review, the definition of the methodology, a detailed description of the data sets that will be used and a plan for evaluating the performance of the proposed method. The data has been analysed and, if necessary, pre-processed so that the experiments can be carried out. Next steps include the execution of the experiments and their evaluation. Some minor additional processing of the data sets and some methodology adjustments can be made along the way, depending on the needs of the project. A detailed plan of the next tasks is attached to this report.

REFERENCES

[1] D. Zhang, Y. Sun, B. Eriksson, and L. Balzano, "Deep unsupervised clustering using mixture of autoencoders," *ArXiv*, 2018.

[2] H. Driver and A. Kroeber, "Quantitative expression of cultural relationships," *University of California Publications in America Archaeology and Ethnology*, 1932.

[3] J. M. et al., "Some methods for classification and analysis of multivariate observations," *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, 1996.

[4] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, 1996.

[5] M. Steinbach, L. Ertoz, and V. Kumar, *The challenges of Clustering High Dimensional Data*. Springer, Berlin, Heidelberg, 2004.

[6] K. Pearson, "On lines and planes of closest fit to systems of points in space," *Philosophical Magazine*, 1901.

[7] J. Xie, R. Girshick, and A. Farhadi, "Unsupervised deep embedding for clustering analysis," *International Conference on Machine Learning*, 2016.

[8] Z. Jiang, Y. Zheng, H. Tan, B. Tang, and H. Zhou, "Variational deep embedding: A generative approach to clustering," *CoRR*, vol. abs/1611.05148, 2016. [Online]. Available: http://arxiv.org/abs/1611.05148

[9] N. Dilokthanakul, P. A. M. Mediano, M. Garnelo, M. C. H. Lee, H. Salimbeni, K. Arulkumaran, and M. Shanahan, "Deep unsupervised clustering with gaussian mixture variational autoencoders," *CoRR*, vol. abs/1611.02648, 2016. [Online]. Available: http://arxiv.org/abs/1611.02648

[10] F. Alimoglu, "Combining multiple classifiers for pen-based handwritten digit recognition," Master's thesis, Institute of Graduate Studies in Science and Engineering, Bogazici University, 1996.

[11] A. Rosenberg and J. Hirschberg, "V-measure: A conditional entropy-based external cluster evaluation measure," *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, 2007.