K-CAE: Image Classification Using Convolutional AutoEncoder Pre-Training and K-means Clustering

Aida Chefrour*[1,2] and Samia Drissi [1,3]

[1] Computer Science Department, Mohamed Cherif Messaadia University,
Souk Ahras, Algeria
[2] LISCO Laboratory, Computer Science Department, Badji Mokhtar University,
B.P-12 Annaba, 23000, Algeria
[3] LiM Laboratory, Faculty of Science and Technology, University of Souk Ahras, P.O. Box 1553, 41000 Souk-Ahras, Algeria
E-mail: aida.chefrour@univ-soukahras.dz, s.drici@univ-soukahras.dz
*Corresponding author

Keywords: deep learning, convolutional autoencoder, K-means

Received: November 8, 2022

The work presented in this paper is in the general framework of classification using deep learning and, more precisely, that of convolutional Autoencoder. In particular, this last proposes an alternative for the processing of high-dimensional data, to facilitate their classification. In this paper, we propose the incorporation of convolutional autoencoders as a general unsupervised learning data dimension reduction method for creating robust and compressed feature representations for better storage and transmission to the classification process to improve K-means performance on image classification tasks. The experimental results on three image databases, MNIST, Fashion-MNIST, and CIFAR-10, show that the proposed method significantly outperforms deep clustering models in terms of clustering quality.

Povzetek: Ta članek predlaga vključitev konvolucijskih samodejnih kodirnikov kot splošne nenadzorovane metode zmanjševanja razsežnosti podatkov za učenje izboljšanja zmogljivosti k-means algoritma pri klasifikaciji slik.

1 Introduction

Computers are a very important part of this society, but there are still so many things that a human does better, despite their limited storage and computational capacity. Probably one of the most intriguing areas of study is learning, which can be described in many ways, including acquiring new knowledge, improving existing knowledge, representing knowledge, organizing knowledge, and discovering facts through experiments. In addition, the continuous growth of data volume contributes to the improvement of techniques that seek the implicit knowledge of these data[1].

Machine learning (ML) is the application of underlying computational methods to experience-based decision-making. It is a very important part of artificial intelligence and should be one of the main characteristics of intelligent systems. By learning, we can exploit and build models of reality based on experiences, either by creating a model completely or by modernizing a partially built model. The goals of machine learning are to provide greater solution accuracy, greater problem coverage, greater economy in obtaining solutions, and greater simplicity in representing knowledge.

Machine learning tasks are divided into three types: supervised, unsupervised, and reinforcement learning. Unsupervised learning or clustering, also known as segmentation, is the grouping into homogeneous classes that consists of representing a cloud of points in any space as a set of groups called clusters. Its goal is to organize a collection of data, examples, and points into clusters (sets) that verify the following statement: Points within the same cluster are more similar and closer to one another than points in different clusters. Today, clustering is a basic and essential preprocessing step for many real-world applications [2].

For example, Machine learning can be used to assist in document analysis, marketing, sales, etc. Specifically, the clustering algorithm can cluster according to various data similarity measures and data clustering patterns to find useful and relevant information for the application. For the samples to be properly allocated to different clusters, the meaningful feature values of the samples must be obtained first. However, in practical applications, the data we obtain is usually large and usually contains noise, which makes clustering a difficult task to perform. For example, in the MNIST dataset, each handwritten digit input image has 784 pixels. Although we know that some pixels (such as pixels in the corners of the image) may not be as useful as other pixels (such as pixels in the center of the image), it is difficult to manually distinguish between them when clustering and to reduce the dimensions and number of features in the cluster. Traditional clustering algorithms can only attain limited performance as the dimensionality increases. Dealing with high-level representation offers beneficial components that make the clustering process possible. Representative features with compact clusters are much more useful because there is no supervision knowledge to provide information about category labels. Unsupervised models for representation learning include convolutional auto-encoders (CAEs). They integrate inputs into a new representation space, allowing the encoding process to provide useful features. The encoding part projects the data into a collection of feature spaces, from which the decoding part reconstructs the original data [3].

In this study, we introduce a clustering method k-means integrated within a CAE framework that aims to simultaneously learn feature representation and cluster assignment. Contrasting traditional clustering techniques, our approach uses deep neural network representation learning to identify compact and representative latent feature spaces for future classification and recognition. We train our model in an end-to-end approach with fixed parameters without any pre-training or fine-tuning techniques, enabling a faster training process. The majority of existing approaches essentially rely on pre-training the parameters using varied values.

The main contributions treated in this paper are:

- A survey of the literature on embedding deep learning and clustering (Section 2).
- A clarification of the deep convolutional autoencoder with embedded clustering (Section 3) was used in the literature review.
- The proposition of the Convolutional Autoencoder (CAE), which is a simple but more general representation learning framework, allows us to reduce the dimension of the database (inputs) and generate a feature vector (minimum dimensional data) before performing the clustering phase by the K-means algorithm (second part) to obtain better results (section 4).
- The experiment that we conducted and the results obtained by our algorithm are presented in section 5.
- A summary of learned lessons and a reflection on future research works (section 6).

2 Related work

Several algorithms for the incorporation of CAE in the K-means exist in the literature. In this section, we outline the best-known and most recent ones. We noticed that all of these algorithms have shown good results in the last few years. However, no one of them could be said to be the best, as they all depend on the content of input parameters and their application domain:

[3] proposed an approach for clustering that is integrated with a deep convolutional auto-encoder (DCAE). Their method simultaneously learns feature representations and cluster assignments through DCAEs, in contrast to conventional clustering approaches. Since DCAEs completely use the capabilities of convolutional neural networks, they are effective for image processing. They use objective functions for clustering and reconstruction. To achieve consistent performance in clustering, all data points are iteratively allocated to their new matching cluster centers during the optimization process. The experimental results on the MNIST dataset demonstrate that, in terms of clustering quality, the proposed method significantly outperforms deep clustering algorithms. We adopt a similar embedding with a convolutional autoencoder based on the k-means clustering algorithm.

[4] presented a novel autoencoder network-based clustering approach. They achieved a stable and compact representation that is better suited for clustering by carefully considering the constraint of the distance between data and cluster centers. They believe that this is the first attempt at creating an auto-encode for clustering. The data can be well partitioned in the altered space since this deep architecture can develop a potent non-linear mapping. The usefulness of the proposed approach has also been shown by the experimental results. Some facts are still inconsistent, however. This problem might be solved by maximizing the difference between cluster centers in the code layer. By contrast, we cover more representation data by CAE than AE.

For the aim of classifying graphs, the authors of [5] proposed the GraphEncoder approach. They start by introducing a deep neural network (DNN), which uses a sparse autoencoder as its basic building block, the normalized graph similarity matrix. The best non-linear graph representations that can rebuild the input matrix and achieve the necessary sparsity attributes are then followed through a greedy layer-wise pretraining approach. The clustering results are obtained by running k-means on the sparse encoding output by the last layer after stacking many layers of sparse autoencoders. In the same way, we were proposing our algorithm, we apply it in the first convolutional autoencoder and the k-means in the second part for clustering.

Recently, in [6], the authors developed a modified deep learning strategy for lung cancer diagnosis that incorporated convolutional neural networks (CNN) with Kernel K-Means clustering. The proposed CNN architecture was used to analyze all of the data in the first step. The kernel k-means clustering approach obtains the attended neurons of the feature map for each image resulting from the convolutional layers in CNN. The centroid of each cluster is then determined using this procedure, which determines the prediction class of each data point in the validation set. Several k values were used in k-fold cross-validation to measure the performance of their suggested strategy.

By combining the K-means clustering technique with deep learning, the authors of [7] present a new idea for image classification. Because there are significant changes in the foreground and background of input images, they use the K-means clustering algorithm for image preprocessing. The accuracy of image preprocessing using K-means clustering can be improved. They use a two-dimensional deep convolutional neural network to categorize images in the BabyAIImage and Question datasets into multiple classes based on shape, color, size, and location. The researchers' goal in this work is to use the deep learning algorithm to create a system that targets children's visual abilities such as visual acuity, tracking, color perception, depth perception, and object recognition. In contrast to our approach, which aims to use K-means for image preprocessing [8] proposed DeepCluster, a clustering algorithm that learns both the parameters of a neural network and the cluster assignments of the generated features. DeepCluster uses a typical clustering technique, k-means, to iterative group the features and uses the following assignments as supervision to update the network's weights. They use DeepCluster to train convolutional neural networks unsupervised on big datasets.

[9] proposed a SARS-CoV-2 population structure based on a convolutional autoencoder (CAE) trained with numerical feature vectors mapped from coronavirus spike peptide sequences to help predict future infection risks. They started by transferring input sequences of spike proteins and reducing their dimensionality into relevant numerical feature vectors suited for clustering. Then, they used principal component analysis (PCA) to produce a projection of a dataset before fitting a model. Using PCA can reduce the input numerical representation of each sample before applying CAE. The proposed method beats K-means and hierarchical clustering. The results show that, in comparison to virus isolates that are more widely distributed, cluster strains provide improved knowledge of the unknown population lineages.

In [10], the FDG PET/CT image features of multiple myeloma (MM) patients were extracted by a convolutional autoencoder. Both supervised and unsupervised clustering of the extracted features allowed significant and independent predictions of worse PFS. The obtained results support the usefulness of AI algorithm-based cluster analyses of FDG PET/CT images for risk stratification of patients with MM.

We would like to address the following points from this brief and selective study of the incorporation of CAE in the K-means:

- CNN remains the most used model in brain tumor classification. We found that the architecture based on the Bayesian capsule neural network gave the lowest value of accuracy (74.4%).
- The preprocessing of the images improves their accuracy. Testing with 10-fold cross-validation improves the results too.
- Most of the previous work has been evaluated using the precision, recall, and F1 score metrics on the data set for better performance evaluation, which is essential to measure the model generalization of the test data.

Table 1 lists several state-of-the-art studies that incorporate CAE into the K-means. The objective of each study is described in the table, along with the classifier model employed, the dataset used, and the performance of the results.

Table 1: Summary of state-of-the-art references.

Ref	**Objective**	**Classifier model**	**Used dataset**	**Performance**

[3]	Learning feature representation and cluster assignment simultaneously.	DCAE	MNIST, USPS	**Accuracy** :
				MNIST=92.14%
				USPS=89.03%

[4]	Learning a highly non-linear mapping function	Objective function embedded into the auto-encoder model	MNIST, USPS, YaleB	**Accuracy** :
				MNIST=76%
				USPS=71.5%
				YaleB=90.2%

[5]	Spectral clustering	SAE: Learning a nonlinear embedding of the graph by stacking an autoencoder and k-means	Wine, 3-NG, 6-NG,9-NG	**Normalization of the Mutual Information**
				NMI-Wine=84%
				NMI_3-NG=81%
				NMI_6-NG=60%
				NMI_9-NG=41%

[6]	Lung cancer diagnosis	Modified deep learning methods combine convolutional neural networks (CNN) and kernel K-means clusterin	-Lung cancer data points, -Healthy lung data points	**Accuracy**=98.85%
[7]	Development of a system that targets the visual abilities of children	2D DCNN	BabyAIImageand Question	**Accuracy**= 96.79%

[8]	Preservation of the local structure of the data-generating distribution by incorporating convolutional layers.	DCEC	MNIST-full, MNIST-test, USPS	**Accuracy :**
				MNIST-full= 88.97%
				MNIST-test=85.29%USPS=79 %

[9]	Identification of the main clusters of SARS-CoV-2 population structure to diagnose it	SARS-CoV-2	Dataset of spike proteins	**Accuracy** = 91.7%
[10]	Identification of multiple myeloma (MM) patients	MTV: an algorithm for feature extraction and prediction based on convolutional autoencoders	Dataset of 254 MM patient	**Statistic Value** P=0.002

3 Background

Our proposed approach comprises two modules: dimension reduction and classification. The dimension reduction is realized using CAE. The classification is carried out using K-means. In this section, we briefly review the main concepts of CAE and K-means.

3.1 Convolutional neural networks (CNN)

CNNs are massively used in image-based learning applications. Due to their autonomous feature extraction technique, CNNs can extract useful data from training samples. CNNs are usually created with several convolutional, pooling, and fully connected layers. To extract features, the input is convolved with convolutional kernels, as shown in Figure 1. Without significantly changing the feature map's resolution, the pooling layer reduces the network's computational complexity. In CNN, as the number of layers increases, the size of the pooling layers typically falls. Max pooling and average pooling are two of the most used forms of pooling layers [11].

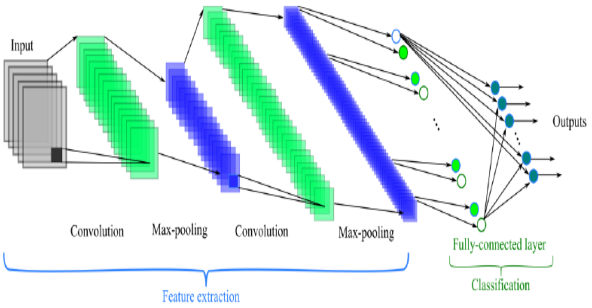


Figure 1: AE architecture [11].

3.2 AutoEncoder (AE)

Although they don't require a training dataset, AEs fit into the category of unsupervised learning. An AE creates a compressed latent space representation of the input data, which then decompresses it to reconstruct the data. In the compression step, AEs carry out dimensionality reduction, which is similar to principal component analysis (PCA) but unlike PCA, which uses linear transformation, AEs use deep neural networks to do the linear transformation.

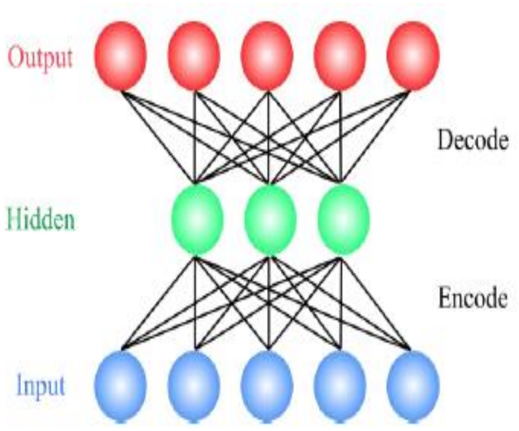


Figure 2: AE architecture [11]

3.3 Convolutional AutoEncoder (CAE)

Convolutional AutoEncoders are unsupervised dimensionality reduction models composed of convolutional layers capable of creating compressed image representations.

In general, CAEs are used to extract robust features, reduce and compress the size of the input dimension, and remove the noise while simultaneously preserving all necessary information.

The use of convolutional layers is the main difference between CAE and traditional AE. It is important to note that these layers are distinguished by their desirable capability of knowledge extraction and internal representation of image data learning.

More specifically, as shown in Figure 3, CAEs are composed of 2 CNN models, the encoder and the decoder. The encoder's principal function is to convert the initial input image into a latent representation with reduced dimensionality. The decoder, on the other hand, is responsible for rebuilding the compressed latent representation and producing an output image that is as similar to the original as possible.

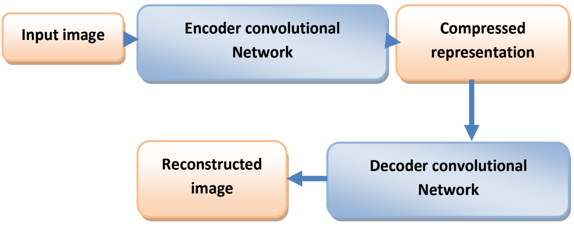

Input image
Encoder convolutional Network
Compressed representation
Decoder convolutional Network
Reconstructed image

Figure 3: CAE architecture.

3.4 K-means clustering algorithm

The most frequently used static clustering method in scientific and industrial applications is the k-means algorithm [12]. It's a clustering approach that divides 'n' observations into k clusters, with each observation belonging to the cluster with the nearest mean (a cluster is represented by its centroid, which is a mean: average).

The basic algorithm is simple, as shown in Figure 4:

```
Input: K the number of clusters to form;
The training set (matrix form);
Start
    • Randomly choose K data (one row of the data matrix). These are the cluster centers (also known as the centroid).
    Repeat
    – Assign each data element (element of the data matrix) to the cluster to whose center it is closest.
    – Recalculate the center of each cluster and modify the centroid.
    Until convergence
    • Or (stabilization of the total inertia of the population)
End algorithm
```

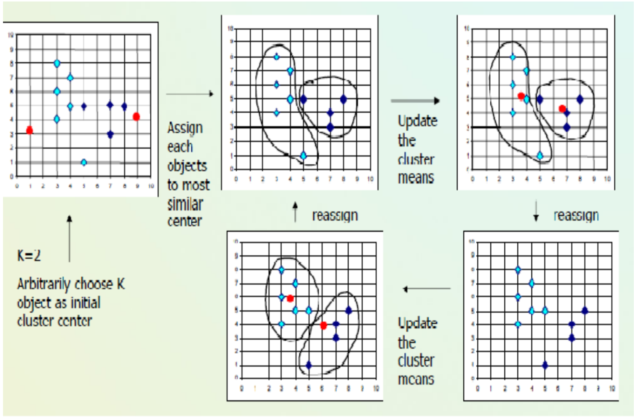

K=2
Arbitrarily choose K object as initial cluster center
Assign each objects to most similar center
Update the cluster means
reassign
reassign
Update the cluster means

Figure 4: K-means working [12]

4 The proposed K-CAE classification algorithm

To overcome the limitations of the data representation and the high dimensionality of the dataset and feature extraction, we have developed in this work an embedding of the CAE and K-means (K-CAE).

The objective of this proposed approach is the application of deep learning to learn models to transform the input data into more user-friendly representations and to reduce the dimensions for classification. The CAE is based on a set of successive transformations that amplify the features of the input data that discriminate against them and attenuate their variations.

The proposed K-CAE architecture is shown in Figure 5. The initial training dataset is used to train a CAE. The decoder component is eliminated once the CAE has completed its training process, and the encoder is employed to reduce the size of the original high-dimensional image dataset into a compressed image dataset. Finally, the compressed image dataset produced by the CAE's encoder is utilized to feed and train a K-means clustering model.

4.1 K-CAE for image recognition

For unsupervised training of CAE, we use three image datasets (MNIST handwritten digit images; Fashion-MNIST, Zalando's article images; and CIFAR-10 Color images) from the UCI machine repository to validate the accuracy and efficiency of our proposed approach.

The MNIST database contains a training set of 60000 examples and a test set of 10000 examples with varying resolutions averaging around 28×28 pixels.

Fashion-MNIST is a dataset comprised of 28×28 grayscale images of 70000 fashion products from 10 categories, with 7000 images per category. The training set has 60000 images and the test set has 10000 images. Fashion-MNIST shares the same image size, data format, and structure of training and testing split with the original MNIST.

The CIFAR-10 dataset consists of 60000 32x32 color images in 10 classes, with 6000 images per class. There are 50000 training images and 10000 test images.

The goal is to train the CAE to find features, but here we use the encoder part for compressing the initial high-dimensional image dataset into a compressed image dataset.

Configuration of model 1 (obtained on the MNIST Dataset):

Our CAE1 contains the following layers:

1. The input layer consists of the raw image (28×28 pixels);
2. A convolutional layer of size 128×28×28;
3. Maxpooling layer of size 2×2;
4. A convolutional layer of size 64×14×14;
5. Maxpooling layer of size 2×2;
6. A convolutional layer of size 32×7×7;
7. Maxpooling layer of size 2×2;
8. Output Encoder of size 32×4×4;
9. Unpooling layer of size 2×2;
10. Deconvolutional layer of size 32×4×4;
11. Unpooling layer of size 2×2;
12. Deconvolutional layer of size 64×8×8;
13. Unpooling layer of size 2×2;
14. Deconvolutional layer of size 128×14×14;
15. Unpooling layer of size 2×2;
16. Deconvolutional layer of size 128×28×28;

After a CAE1 has been trained, the decoder components (items 9 to 16 in the list above) can be removed, and the CAE can then be used to initialize unsupervised K-means. The softmax activation function is applied.

Configuration of model 2 (obtained on the Fashion-MNIST Dataset):

We obtained the same results as the MNIST dataset.

Configuration of model 3 (obtained on the CIFAR-10 Dataset):

Our CAE3 contains the following layers:

1. The input layer consists of the raw image (32×32 pixels);
2. A convolutional layer of size 64×32×32;
3. Maxpooling layer of size 4×4;
4. A convolutional layer of size 32×16×16;
5. Maxpooling layer of size 4×4;
6. A convolutional layer of size 16×8×8;
7. Output Encoder of size 16×8×8;
8. Unpooling layer of size 4×4;
9. Deconvolutional layer of size 32×8×8;
10. Unpooling layer of size 4×4;
11. Deconvolutional layer of size 64×16×16;
12. Unpooling layer of size 4×4;
13. Deconvolutional layer of size 64×32×32;

After a CAE3 has been trained, the decoder components (items 8 to 13 in the list above) can be removed, and the CAE can then be used to initialize unsupervised K-means. The softmax activation function is applied.

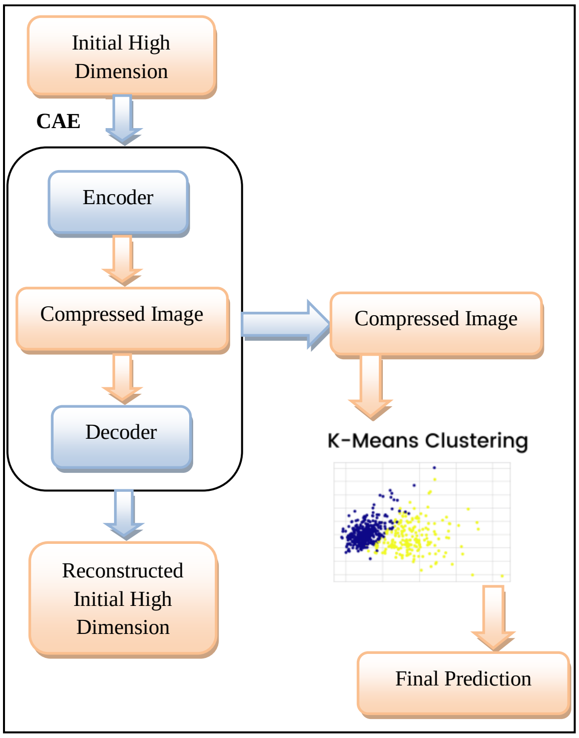


Figure 5: The proposed architecture of K-CAE

5 Experimental results

5.1 Performance evaluation parameters

In this subsection, we validate the efficiency and robustness of the proposed approach by performing comprehensive experimental simulations. The measurement of quality is based on the well-known and widely used evaluation metrics: accuracy (Acc), precision, recall, and F1-score. These parameters can be calculated using Equations (1, 2, 3, and 4):

$$\text{Acc}=\frac{TP+TN}{TP+FP+TN+FN} \quad (1)$$

$$\text{Precision}=\frac{TP}{TP+FP} \quad (2)$$

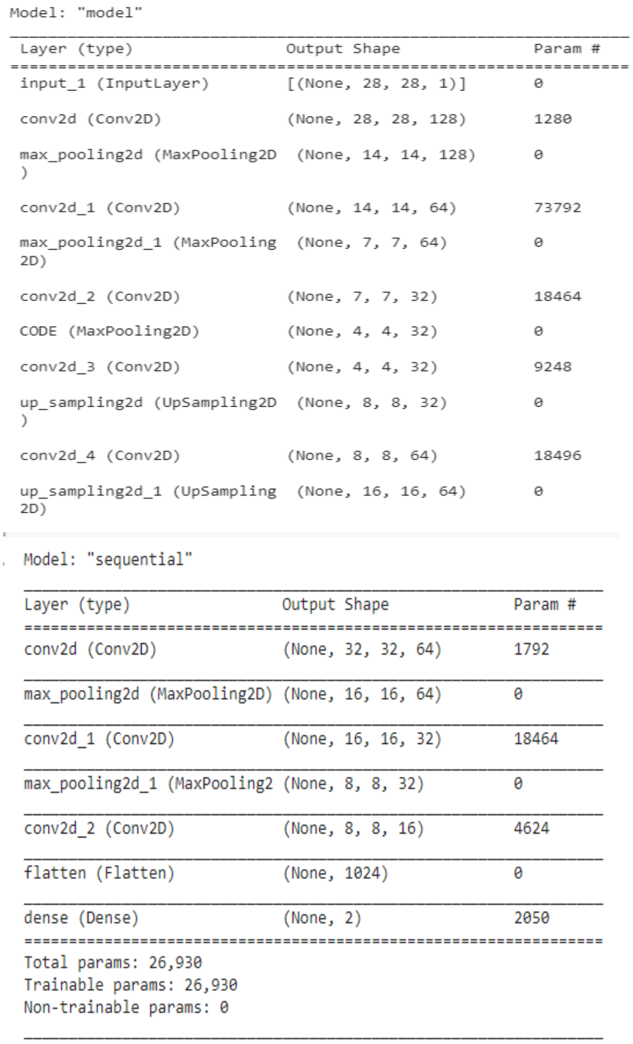
$$\text{Recall}=\frac{TP}{TP+FN} \quad (3)$$

$$\text{F1\_score}=\frac{2\times Precision\times Recall}{Precision+Recall} \quad (4)$$

5.2 Results

The classification process consists of two steps: the first one performs the dimension reduction with CAE, and the second stage represents the decision-making process for classification using K-means.

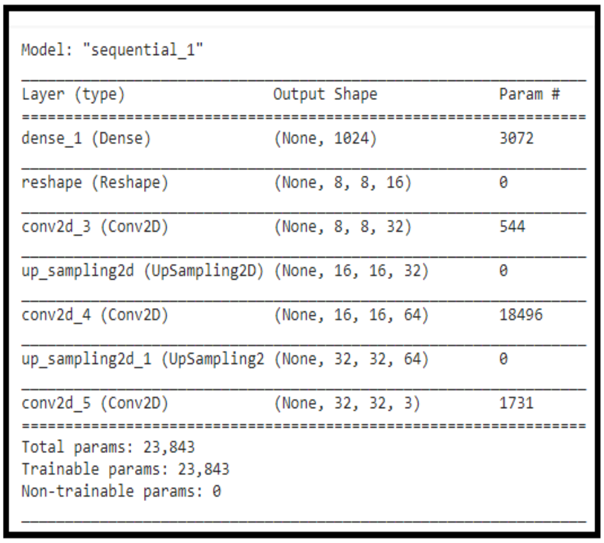
We train CAE on the three datasets. We obtain the following models, respectively (as shown in Figures 6 and 7):



```
Model: "model"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_1 (InputLayer)        [(None, 28, 28, 1)]       0

 conv2d (Conv2D)             (None, 28, 28, 128)       1280

 max_pooling2d (MaxPooling2D  (None, 14, 14, 128)      0
 )

 conv2d_1 (Conv2D)           (None, 14, 14, 64)        73792

 max_pooling2d_1 (MaxPooling  (None, 7, 7, 64)         0
 2D)

 conv2d_2 (Conv2D)           (None, 7, 7, 32)          18464

 CODE (MaxPooling2D)         (None, 4, 4, 32)          0

 conv2d_3 (Conv2D)           (None, 4, 4, 32)          9248

 up_sampling2d (UpSampling2D  (None, 8, 8, 32)         0
 )

 conv2d_4 (Conv2D)           (None, 8, 8, 64)          18496

 up_sampling2d_1 (UpSampling  (None, 16, 16, 64)       0
 2D)
```

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 32, 32, 64)        1792
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 16, 16, 64)        0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 16, 16, 32)        18464
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 8, 8, 32)          0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 8, 8, 16)          4624
_________________________________________________________________
flatten (Flatten)            (None, 1024)              0
_________________________________________________________________
dense (Dense)                (None, 2)                 2050
=================================================================
Total params: 26,930
Trainable params: 26,930
Non-trainable params: 0
_________________________________________________________________
```

Figure 6: Illustration of CAE (MNIST and Fashion-MNIST databases)



```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_1 (Dense)              (None, 1024)              3072
_________________________________________________________________
reshape (Reshape)            (None, 8, 8, 16)          0
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 8, 8, 32)          544
_________________________________________________________________
up_sampling2d (UpSampling2D) (None, 16, 16, 32)        0
_________________________________________________________________
conv2d_4 (Conv2D)            (None, 16, 16, 64)        18496
_________________________________________________________________
up_sampling2d_1 (UpSampling2 (None, 32, 32, 64)        0
_________________________________________________________________
conv2d_5 (Conv2D)            (None, 32, 32, 3)         1731
=================================================================
Total params: 23,843
Trainable params: 23,843
Non-trainable params: 0
_________________________________________________________________
```

Figure 7: Illustration of CAE (CIFAR-10 database)

Obtained results for models 1 and 2:

In the first part, the model generated by CAE and presented in Figure 6 is composed of seven convolution layers, three max pooling layers, and two fully connected layers.

The input image is of size 28×28. It goes first to the convolution layer, which is composed of 14 filters. Each of our layers of convolution is followed by a function of activation called RELU, which forces the neurons to return positive values.

The outputs of the CAE1 and 2 are a reduced-size image of 4×4.

In the training process of CAE1 and 2, the data is divided into training and test sets. Therefore, 60000 samples were used to train the CAE model, and the remaining 10000 samples were used for testing purposes to calculate the accuracy error (as described above). We obtain an accuracy of 81.44% after training the CAE 1 and 2 models for 50 epochs. There is still a modest result.

To improve this result:

In the second part, we apply the K-means clustering algorithm (see Figure 8) to the results of the encoder and the feature vector resulting from the previous step to determine which of the images are similar to each other and group them into one of the 10 classes. The division of the database into learning and testing remains the same. We obtain an accuracy of 96.22% after training the CAE1 and 2 models for 50 epochs.

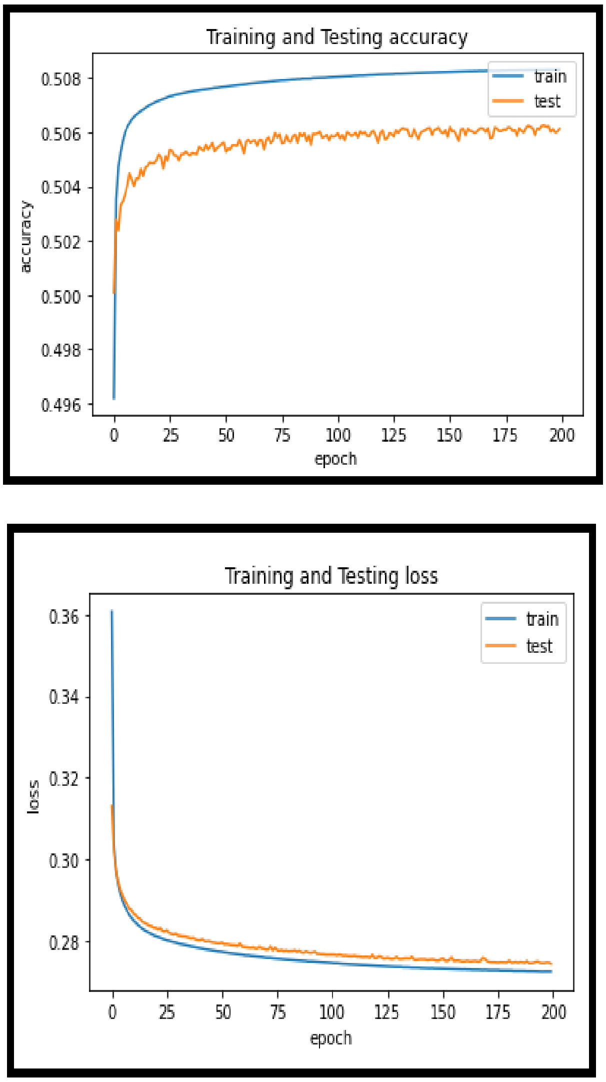


Figure 8: Accuracy and error rates obtained from K-CAE (MNIST and Fashion-MNIST databases)

Obtained results for model 3:

In the first part, the model generated by CAE and presented in Figure 7 is composed of five convolution layers, two maxpooling layers, and one fully connected layer.

The input image is of size 32×32. It goes first to the convolution layer, which is composed of 16 filters. Each of our layers of convolution is followed by a function of activation called RELU, which forces the neurons to return positive values.

The output of the CAE3 is a reduced-size image of 8×8.

In the training process of CAE3, the data is divided into training and test sets. Therefore, 50000 samples were used to train the CAE model, and the remaining 10000 samples were used for testing purposes to calculate the accuracy error (as described above). We obtain an accuracy of 63.95% after training the CAE 3 model for 50 epochs. Still, the result is modest.

To improve this result:

Table 2: The results of the K-CAE (MNIST database).

Models	N. of epochs	Architecture of CAE			Acc (%)	Error (%)
		N. of convolution layers	N.of pooling layers	N.of connec ted layers		
Model 1 (MNIST DB)	50	7	3	2	81.4	96.22
Model 2 (Fashion-MNIST DB)	50	7	3	2	81.4	96.22
Model 3 (CIFAR-10 DB)	50	5	2	1	63.9	76.48

In the second part, we apply the K-means clustering algorithm (see Figure 9) to the results of the encoder and the feature vector resulting from the previous step to determine which of the images are similar to each other and group them into one of the 10 classes. The division of the database into learning and testing remains the same. We obtain an accuracy of 76.48% after training the CAE3 model for 50 epochs.

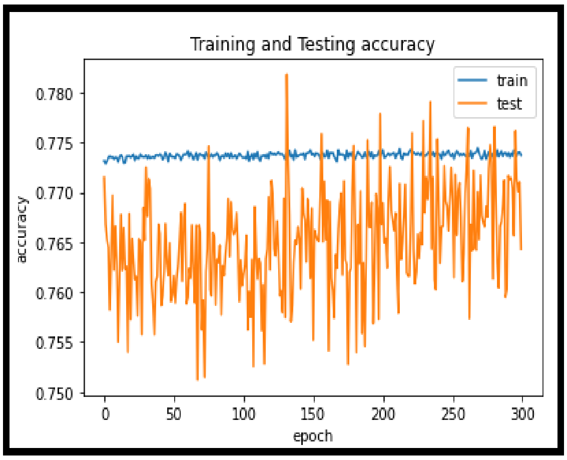


Figure 9: Accuracy obtained from K-CAE (CIFAR-database)

We discovered (see Figures 8 and 9) that the number of epochs (epoch=50) increases the accuracy of learning and testing. The results found are good, and the models learn more information. In contrast, as the number of epochs increases, the error (loss) of learning and testing decreases.

5.3 Discussion

In this study, we have shown that the incorporation of convolutional autoencoders as an image preprocessing technique (dimension reduction) could improve the performance of K-means models, leading to robust and accurate results. Therefore, it can be considered a promising tool for high-dimensional and noisy dataset applications.

Table 2 summarizes the performance of the proposed approach regarding the MNIST dataset for the 10 classes in terms of the evaluation performance measures.

Table 3: Comparison of the results of our proposed three models.

Classes	Precision	Recall	F1_score	Accuracy
0	0.98	0.99	0.98	**0.96**
1	0.99	0.98	0.99	
2	0.96	0.98	0.97	
3	0.97	0.95	0.96	
4	0.98	0.96	0.97	
5	0.99	0.93	0.96	
6	0.99	0.96	0.98	
7	0.99	0.93	0.96	
8	0.84	0.99	0.91	
9	0.95	0.94	0.95	

Table 3 compares the results obtained by our three proposed models according to the following criteria: number of epochs, CAE architecture (number of convolution layers, number of pooling layers, and number of fully connected layers), accuracy rate, as well as error rate.

- We noticed that the first and second models (applied to the MNIST and Fashion- MNIST databases) gave the same and good results compared to the third model (applied to the CIFAR-10 database), and this is due to the number of convolution layers and the number of intermediate pooling layers. As this number increases, the performance increases.

- According to our study, we noticed that the results obtained by the application of CAE are close to the results obtained after the integration of the k-means clustering method in the deep classification by CAE and sometimes better.

We demonstrate the effectiveness of our K-CAE algorithm mainly by comparing it with the deep convolutional embedded clustering method (DCAE)[3] and the AutoEncoder Clustering (AEC) [4] algorithm, in terms of accuracy, and we evaluated it on the MNIST dataset.

To validate the performance of our proposed method, We compared our method with five baseline methods: K-means, CAE, AEC [4], DCAE [3], and SARS-CoV-2 [9]. The results are summarized in Table. 4. Our proposed method outperforms the baseline methods by a significant margin in accuracy (96.22%). Especially, the proposed method substantially outperforms the second-place method by 3.52%, which also uses the CAE approach with jointed clustering loss.

Table 4: The results of the evaluation parameters of six different approaches.

Methods	**Accuracy**
K-means	57%
CAE	81.44%
AEC [4]	76.00%
DCAE [3]	92.14%
SARS-CoV-2[9]	91.7%
Proposed K-CAE	**96.22%**

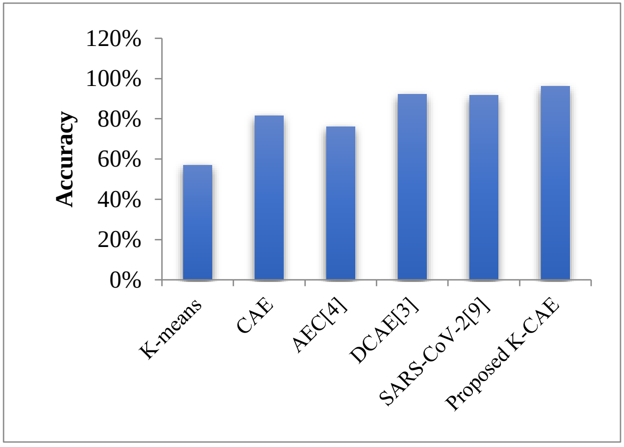


Figure 10: Accuracy comparison across all state-of-the-art studies.

Sometimes, the dimensionality of the input data is very high, and classical learning algorithms cannot provide better performance. To overcome this problem, deep learning algorithms can reduce the dimensionality of the data, such as convolutional neural networks based on the multilayer perceptron.

In this work, we proposed and suggested the incorporation of convolutional autoencoders as a general unsupervised learning data dimension reduction method for creating robust and compressed feature representations to improve K-means performance on image classification tasks.

The results presented in this paper show that deep learning methods can be effectively employed for image classification. Our results show that CAEs are capable of extracting meaningful information from digits by dimension reduction, and when combined with the K-means clustering algorithm, we were able to significantly improve classification accuracy.

Our work opens the way to many perspectives that can be incorporated in the future. Among which we can cite:

- We will also use other supervised deep learning algorithms, such as Convolutional Neural Networks (CNN), etc [13];
- With the use of large data sets, we will introduce the notion of incrementality into the database provided to the autoencoder [14];
- This architecture can also be used in certain application domains, such as handwriting recognition, with very large datasets.

Acknowledgment

The authors would like to thank the DGRSDT (General Directorate of Scientific Research and Technological Development) - MESRS (Ministry of Higher Education and Scientific Research), ALGERIA, for the financial support of LISCO Laboratory, and the LiM Laboratory.

References

[1] JOST, Ingo et VALIATI, Joao Francisco. Deep Learning Applied on Refined Opinion Review Datasets. *Inteligencia Artificial*, 21:91-102,2018. https://doi.org/10.4114/intartif.vol21iss62pp91-102

[2] Abudalfa, Shadi, and Mohammad Mikki. K-means algorithm with a novel distance measure. *Turkish Journal of Electrical Engineering and Computer Sciences*, 21(6):1665-1684, 2013. doi:10.3906/ELK-1010-869

[3] ALQAHTANI, Ali, XIE, Xianghua, DENG, Jingjing, et al. A deep convolutional auto-encoder with embedded clustering. In *25th IEEE international conference on image processing (ICIP)*, 4058-4062, 2018.

[4] SONG, Chunfeng, LIU, Feng, HUANG, Yongzhen, et al. Auto-encoder based data clustering. In *Iberoamerican congress on pattern recognition*, 117-124, 2013.

[5] TIAN, Fei, GAO, Bin, CUI, Qing, et al. Learning deep representations for graph clustering. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 28(1), 2014.

[6] RUSTAM, Zuherman, HARTINI, Sri, PRATAMA, Rivan Y., et al. Analysis of architecture combining convolutional neural network (CNN) and kernel K-means clustering for lung cancer diagnosis. *Int. J. Adv. Sci. Eng. Inf. Technol*, 10(3):1200-1206, 2020. doi:10.18517/ijaseit.10.3.12113

[7] MAW, Swe Zar, ZIN, Thi Thi, YOKOTA, Mitsuhiro, et al. Classification of shape images using K-means clustering and deep learning. *ICIC Express Letters*, 12(10): 1017-1023,2018. doi: 10.24507/icicel.12.10.1017

[8] GUO, Xifeng, LIU, Xinwang, ZHU, En, et al. Deep clustering with convolutional autoencoders. In *Proceedings of the International conference on neural information processing, Springer, Cham*, 373-382, 2017.

[9] Sherif, Fayroz F., and Khaled S. Ahmed. Unsupervised clustering of SARS-CoV-2 using deep convolutional autoencoder. *Journal of Engineering and Applied Science*, 69(1): 72, 2022. Doi:10.1186/s44147-022-00125-

[10] LEE, Hyunjong, HYUN, Seung Hyup, CHO, Young Seok, et al. Cluster analysis of autoencoder-extracted FDG PET/CT features identifies multiple myeloma patients with poor prognosis. *Scientific Reports*, 13(1):7881, 2023. Doi:10.1038/s41598-023-34653-3

[11] KHOZEIMEH, Fahime, SHARIFRAZI, Danial, IZADI, Navid Hoseini, et al. Combining a convolutional neural network with autoencoders to predict the survival chance of COVID-19 patients. *Scientific Reports*, 11(1):1-18, 2021. doi: 10.1038/s41598-021-93543-8

[12] JUNIOR, João Batista Pacheco et DO AMARAL, Henrique Mariano Costa. Performance Analysis in the Segmentation of urban asphalted roads in RGB satellite images using K-Means++ and SegNet: Case study in São Luís-MA. *Inteligencia Artificial*, 24(68):89-103, 2021. doi: 10.4114/intartif.vol24iss68pp89-103

[13] Chefrour, Aida, and Labiba Souici-Meslati. Unsupervised Deep Learning: Taxonomy and algorithms. *Informatica*, 46(2):151-168, 2022. https://doi.org/10.31449/inf.v46i2.3820

[14] Chefrour, Aida, and Labiba Souici-Meslati. AMF-IDBSCAN: Incremental density based clustering algorithm using adaptive median filtering technique. *Informatica*, 43(4):495-506, 2021. https://doi.org/10.31449/inf.v43i4.2629