Information Sciences 565 (2021) 146–176

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

ClEnDAE: A classifier based on ensembles with built-in dimensionality reduction through denoising autoencoders

Francisco J. Pulgar *, Francisco Charte, Antonio J. Rivera, María J. del Jesus

Andalusian Research Institute on Data Science and Computational Intelligence (DaSCI), Computer Science Dpt., University of Jaén, 23071 - Jaén, Spain

ARTICLE INFO

Article history:
Received 3 October 2019
Received in revised form 16 February 2021
Accepted 18 February 2021
Available online 27 February 2021

Keywords:
Classification
Deep learning
Denoising autoencoders
Dimensionality reduction
Ensembles
feature fusion

ABSTRACT

High dimensionality is an issue that affects most classification algorithms. This factor implies that the predictive performance of many traditional classifiers decreases considerably as the number of features increases. Therefore, there are numerous proposals that try to mitigate the effects of this issue. This study proposes ClEnDAE, a new classifier based on ensembles whose components incorporate denoising autoencoders (DAEs) to reduce the dimensionality of the input space. On the one hand, the use of ensembles improves the predictive performance by using several components that work jointly. On the other hand, the use of DAEs allows a new higher-level, smaller-sized feature space to be generated, reducing high dimensionality effects. Finally, an experimentation is conducted with the goal of evaluating the behavior of ClEnDAE. The first part of the test compares the performance of ClEnDAE to a model based on basic DAE and to the original untreated data. The second part analyzes the results of ClEnDAE and other traditional methods of dimensionality reduction in order to determine the improvement achieved with the proposed algorithm. In both parts of the experimentation, conclusions show that ClEnDAE offers better predictive performance than the other analyzed models. The main advantage of the ClEnDAE method is the combination of the potential of the ensemble-based methodology, where several components work in parallel, and DAEs, which generate new low-dimensional features that provide more relevant information. Therefore, the classification performance is better than with other classic proposals.

1. Introduction

Classification is one of the best known and most studied tasks in machine learning. Essentially, a classifier analyzes a series of training instances to extract relevant information that allows it to predict properties of new patterns. Over the years, a large number of proposals that deal with this problem have emerged. These methods are based on different methodologies associated with their structure and internal functioning [1].

The wide use of classification methods that are applied to very diverse data causes new problems and challenges associated with the properties of these data. One of the issues that affects most of the classification systems is the high dimensionality of the aforementioned data [2]. In recent years, this factor has increased considerably due to the continuous evolution of information collection systems. At present, a large number of tools allow huge amounts of information to be generated, stored and managed, making the task of analyzing them more relevant as a result. In this context, the perfor-

* Corresponding author.
E-mail addresses: fpulgar@ujaen.es (F.J. Pulgar), fcharte@ujaen.es (F. Charte), arivera@ujaen.es (A.J. Rivera), mjjesus@ujaen.es (M.J. del Jesus).

https://doi.org/10.1016/j.ins.2021.02.060
0020-0255/

mance of a significant number of traditional classification algorithms is negatively affected by the high dimensionality of the data [3]. The main reason for this behavior lies in the curse of dimensionality [4]. Fundamentally, this phenomenon is related to the decrease in performance of traditional classifiers as the dimension of the input data increases [5]. Hence, developing new models or adapting traditional algorithms to deal with this problem is crucial.

In regard to this matter, some proposals address data dimensionality reduction. Essentially, the goal is to reduce dimensionality while preserving most of the relevant information. In this way, classifier performance improves when working with fewer input features [6]. Some of the most well-known traditional dimensionality reduction methods are: Principal Component Analysis (PCA) [7], Linear Discriminant Analysis (LDA) [8], Isometric feature mapping (ISOMAP) [9] and Locally Linear Embedding (LLE) [10], among others [11].

Lately, a new methodology, known as Feature Fusion, has emerged [12]. The constant growth in the dimensionality of the data and the need to work with multimedia data have influenced the appearance of this new concept [13]. The main objective of Feature Fusion is to generate new features based on the information provided by the whole set of original characteristics. In this manner, relevant input information is used and redundant or irrelevant data is discarded. Some of the most popular methods that apply Feature Fusion are based on Deep Learning (DL) [14]. For instance, Convolutional Neural Networks (CNNs) [15] are able to identify complex patterns in images, and Autoencoders (AEs) [16] concentrate all the information needed to reconstruct the patterns in a few high-level features. These types of algorithms have proven to offer good results in many fields of application, which implies that their use has significantly increased [17].

In relation to dimensionality reduction, AEs are the most used models due to their method of operation [16]. In fact, there are several studies that show the improvement of predictive performance and execution time when using AEs to reduce input data dimensionality [12,18,19]. The AEs rely on nonlinear relationships of the input data to obtain a new representation. This is the main distinguishing feature when comparing with traditional methods such as PCA, LDA, LLE or ISOMAP, which are based on input features linear relationships.

In addition, using ensembles implies the integration of a set of components that solve the same task in a single predictive model. Some research shows that these components offer better results when working together than when used in isolation [20,21]. Furthermore, there are other studies that relate the use of ensembles to DAEs [22]. These reasons, among others further extended in Section 2.5, justify the use of ensembles in this proposal.

Specifically, the Classifier Ensemble DAE (ClEnDAE) model is proposed in this work. This algorithm is based on an ensemble of Denoising AEs (DAEs) to reduce the dimensionality of the input space. The new variables generated by ClEnDAE in the feature fusion phase can be used as a new representation of the input data and serve as input for traditional data mining methodologies. In the second phase of ClEnDAE, different algorithms can be used to classify. Concretely, the possible classifiers are k-Nearest Neighbors (kNN) [23], Support Vector Machines (SVM) [24], Multi-Layer Perceptron (MLP) [25] and C4.5 [26]. The ClEnDAE method is parameterized so as to select the desired classifier from the previous options. Based on the results of these methods, it is possible to evaluate the quality of the features generated by the proposed model. Essentially, classifiers are applied using the new representation of the input generated by the different components of the ensemble. Finally, the information provided by them is combined to generate the final response of the model. Fig. 1 shows this operation schematically.

In summary, ClEnDAE merges the information provided by a certain number of models in order to classify new data. This method addresses dimensionality reduction from different perspectives. Firstly, each component is trained with a subset of the original examples and features randomly sampled with replacement and a new representation is then obtained through the use of DAEs. This process allows each model to face dimensionality reduction using DAEs before classifying with four classical algorithms: kNN, SVM, MLP and C4.5.

More precisely, the main contributions of this paper are: 1) the development of a new classification model, ClEnDAE, based on ensembles and DAEs to reduce input dimensionality, 2) a thorough experimentation to verify the behavior of the proposed model against traditional methods and a model based on basic DAE, 3) a comparison between ClEnDAE and four classic models of dimensionality reduction, such as PCA, LDA, ISOMAP and LLE, and 4) a general conclusion on how to use and operate the proposed algorithm depending on the characteristics of the input data.

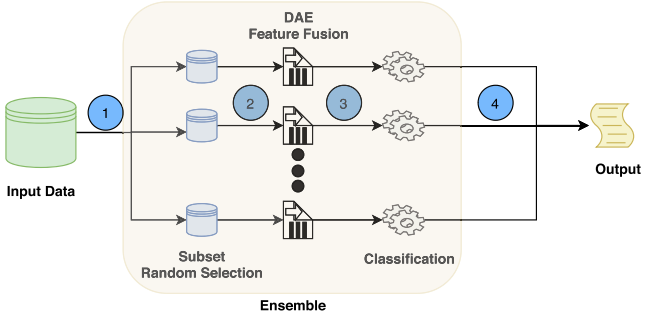

Fig. 1. ClEnDAE scheme.

Lastly, the experimentation developed to verify the proper functioning of the ClEnDAE algorithm, based on 20 datasets, shows significant improvements over traditional classification methods. Additionally, ClEnDAE achieves better performance in most analyzed cases than traditional dimensionality reduction algorithms, for instance, PCA, LDA, ISOMAP and LLE. These data show the value of ClEnDAE when facing the task of dimensionality reduction, considerably improving predictive performance if compared to classical methods.

In summary, the main advantage of this method is that it combines the power of ensemble-based methodology and DAEs that generate new characteristics from the input data with the most relevant information to, subsequently, use that information to carry out the classification. Therefore, the performance of this method is better than that of other traditional proposals.

The article is organized as follows. In Section 2, main theoretical concepts related to the model developed in this paper are presented. Section 3 details the proposed ClEnDAE method. In Section 4, the experimental framework is presented and the obtained results are analyzed. Finally, Section 5 presents the general conclusions of this study.

2. Preliminaries

This section introduces all theoretical concepts related to ClEnDAE. SubSection 2.1 describes the problem of high dimensionality and some traditional proposals that tackle it. SubSection 2.2 presents the traditional classifiers that have been applied, as well as the family of algorithms they belong to. In addition, the concept of AE and its inner workings are analyzed in SubSection 2.3. The AE model used in this paper, DAE, is outlined in SubSection 2.4. Finally, the ensemble methodology is detailed in Section 2.5.

2.1. Dimensionality reduction methods

A common characteristic of any classification methodology, like the four presented in Section 2.2, is that its performance is hindered by input data complexity. These algorithms generally use data from very diverse sources and varied scopes. Lately, growing dimensionality is a common factor in these data [27]. This property is one of the traits that negatively affects predictive performance in many classifiers, due to the phenomenon known as the curse of dimensionality. This factor refers to the decrease in classifier performance from a certain high number of features onward. For this reason, it is necessary to generate a new feature space of lower dimensionality, by selecting the most relevant attributes or by creating new features that aggregate the most important information.

Furthermore, another related effect implies that a large number of redundant features in a high dimensional space will impact the performance of a classifier. This will occur when using a limited number of training instances, and is known as the Hughes phenomenon. In many cases, models need to increase the number of training examples to generate acceptable classification results. This subsection presents some proposals that try to mitigate these effects.

There are different paradigms that address the dimensionality reduction task. Fig. 2 summarizes the paradigms described below. Here are some of the most important:

- Feature selection: This type of method selects a subset of features of the input data. This process does not modify the original features, but rather selects those considered most relevant. The disadvantage of some basic methods is that they treat variables independently, without taking into account information that is provided by the complete set [28].

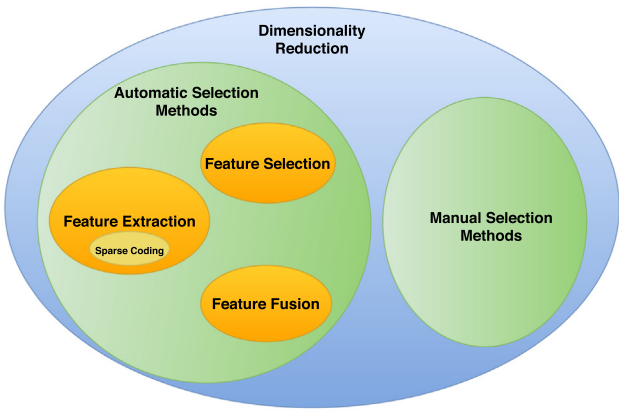

Fig. 2. Dimensionality reduction methods.

- Feature extraction: These algorithms try to generate a new representation for input data with lower dimensionality [29]. Doing so usually involves performing linear data transformations. This technique will depend on the input data, as well as the subsequent use of the generated information. Some of the most popular methods within this methodology are LDA, PCA, ISOMAP and LLE. Another technique which can carry out this task is sparse dictionary learning or sparse coding. These methods aim to generate a new sparse representation of the input data by using linear combinations of them. This methodology is applied in many fields, compression sensing being one of the most important.
- Feature fusion: The increasing use of multimedia data in recent years has led to the emergence of a new dimensionality reduction approach, known as Feature Fusion [13]. The main objective is to generate a new space of features that contains the most relevant information of the input space, discarding irrelevant and unuseful information. Therefore, the use of new features in later stages improves the performance of models such as classifiers. In this context, one of the most popular techniques to develop algorithms is DL, specifically AEs.

The aforementioned methods that address the dimensionality reduction task are used in a wide variety of application domains. Some of them are: cyber security, data visualization, time series, big data analytics and real time decision support systems, among others.

As indicated before, there are a large number of models that tackle dimensionality reduction. However, AE-based feature fusion models have proven to be very efficient and offer good performance [3,12]. This is why the ClEnDAE model is based on DAE, a specific type of AE, in orderto mitigate the effects of high dimensionality. SubSection 2.3 introduces the concept of AE and SubSection 2.4 presents the term DAE.

2.2. Classifiers

Classification is among the most important tasks in the machine learning field. From four of the best known and most widely used paradigms, four classifiers will be considered later on. Nonetheless, there are many other types of methods, for example, rule-based systems, deductive logic programming and genetic algorithms, among others. Next, the four families of classifiers and the selected methods are described. Fig. 3 shows a taxonomy of these methods.

- Instance-Based Learning (IBL): The main characteristic of this type of methods is that they do not build a model from the training data, that is, it is a *lazy* paradigm. These algorithms use the information provided by the training instances to directly infer the prediction corresponding to each new example. Normally, these algorithms base the response on a specific subset of the training space that is related to the new instance [30].
  In this context, one of the most widespread algorithms is kNN. It is a nonparametric method used for different types of tasks, including classification or regression [23]. kNN is a *lazy* method that, as stated above, does not need to build a model to generate the prediction [31]. Instead, kNN uses the information provided by the k instances closest to the new example. The distances are calculated in the n-dimensional space generated by the input features. This way, the output provided will be the most common among the neighboring k.

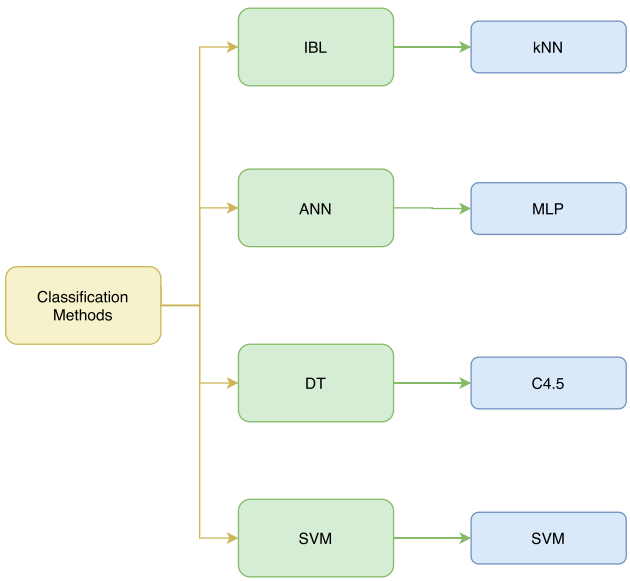

Fig. 3. Classification methods.

- Support Vector Machines (SVMs): These methods are based on the spatial representation of the instances. In particular, a relationship between the different instances and points of space is established. Once the instances are located in the space, the process consists in determining the support vectors of the hyperplane that achieves a greater separation between the represented points. Thus, the elements of different classes are separated in space and the classification of new examples only consists in determining which set is the closest [32].
  The SVM algorithm was introduced in 1992 [24] hland is commonly used due to its good performance and robustness. During the learning process, the method projects the instances in a space and then determines the hyperplane that offers the greatest separation between them. Lastly, the prediction is based on spatial separation. On certain occasions, establishing a separation between the examples is infeasible. In these cases, the algorithm uses nonlinear mapping techniques on the original feature space [32].
- Artificial Neural Networks (ANNs): These models are rooted on the functioning of the human brain. The structure of the model is built out of different layers composed of a variable number of elements, called neurons. These elements are related to each other through different types of connections with weights. It is important to note that these models use nonlinear transformations on the input to determine the output, so both weights and nonlinear transformations are relevant. During the training process, ANNs learn about high-level relationships between the data by themselves. This allows them to make predictions when they receive unlabeled data [33–35].
  MLP [25] has been used to evaluate the ANNs in this study, the main reason being that it is a simple model which offers good classification performance. MLP allows knowledge to be generalized and extracted from the input data. Throughout the learning process, the model adjusts the weights according to the data problem. Finally, the model generates an output vector which is the result of mapping the input data within the network [36].
- Decision Trees (DTs): The most identifying feature of this type of model is its tree-based structure. The fundamental parts of these models are branches and leaves. Branches allow to evaluate the attributes of the data that determine a greater separation of the examples in different categories. The leaves determine the value of the target class. During the prediction process, the model will evaluate each of the attributes of the instance (branches) until it reaches a target class value (leave) [26]. There are a large number of tree-based methods that can be applied to classification. One of the classic methods within this paradigm is C4.5. This algorithm is frequently used for classification. Fundamentally, branches allow the different features to be evaluated, which guides the process towards one of the tree leaves that contains the value of the objective class. Particularly, C4.5 generates a series of classification rules to make new predictions [26].

High dimensionality affects the four classification paradigms presented above. However, the reasons for the decrease in performance are different in each of them. The IBL and SVM methods ground their operation on distances between the input examples. In spaces of high dimensionality, the distances tend to equalize, so the predictive performance decreases noticeably. IBL algorithms using less significant distances cannot correctly identify the nearest neighbors, which implies worse performance. For its part, SVM cannot establish valuable relationships between instances using distances that are not significant. Similarly, high dimensionality negatively impacts ANNs. The use of more features prevents extracting relevant high-level relationships, since, in many cases, it involves irrelevant or redundant information. Lastly, decision trees produce gigantic structures when they work with high dimensional data, meaning that the evaluation time of each of the nodes is very high and the computational performance of the model decreases.

For these reasons, the four aforementioned algorithms have been selected for this study. kNN, SVM, MLP and C4.5 will be used to evaluate the performance of ClEnDAE. This way, the behavior of the proposed algorithm is analyzed from different perspectives. This helps obtain a much more general vision as well as draw conclusions about the effects of dimensionality reduction with the ClEnDAE method when applying four different classification methodologies. SubSection 3 presents the ClEnDAE method and Section 4 details the experimentation conducted to evaluate it.

2.3. Autoencoder foundations

The dimensionality reduction model proposed in this paper is based on the use of a type of AE. An AE is a kind of ANN that has a series of specific characteristics. Its main purpose is to reproduce the input of the network at the output [16]. The model uses exclusively the input attributes to learn an internal representation that allows it to produce the appropriate output, i.e., it is an unsupervised learning method. To achieve this goal, AEs have a specific structure that also prevents the network from merely copying the input [12,18].

AEs have a symmetric structure. There is no doubt that the input and output layers must have the same size, since their objective is to reproduce the input in the output. However, not only these two layers are symmetric, but the entire network is. This structure allows the network to learn an internal encoding from which it can reconstruct the input.

This study focuses on undercomplete AEs considering that, thanks to their architecture, they are the most suited to reducing the dimensionality of the input space. AEs of this kind generate an encoding in their hidden layer by combining the most relevant information of the input and discarding redundant or unuseful content. In this manner, this new representation of the input information constitutes a new space of characteristics of lower dimensionality, which are obtained from the original features [16].

The minimum architecture of an AE contains an input layer, a hidden layer, and an output layer. This architecture can be more complex by adding extra hidden layers. Even so, the structure must always be symmetric according to the characteristics of the AEs [18]. Fig. 4 presents an example of the basic architecture of an AE.

Fig. 4 shows that the AE is composed of two fundamental blocks. The *encoder*, where the model learns the coding of the input, and the *decoder*, where network reconstructs the input into the output from the coding. Another important aspect is that each neuron is connected to the whole next layer through a series of weights, as denoted by W_i in the figure.

The architecture described in this Section corresponds to the most basic AE model. However, there are different types of AEs with more complex functionings and some of them are: robust AE [37], contractive AE [38] and DAE [39]. Section 2.4 details the DAE model, since it is the basis of the proposal of this study.

2.4. Denoising Autoencoder

AEs are useful models to address feature fusion. Nonetheless, this type of network can be limited to copying input information, namely, learning the identity function. This occurs especially when the number of nodes in the hidden layer exceeds the amount of input nodes. This situation renders the work of the AE useless.

In this scenario, the reconstruction method by itself is not capable of guaranteeing the generation of useful features. There are different strategies that can be applied to deal with this problem. For instance, by constraining the original data: bottleneck or sparse representations of the input. Furthermore, another alternative involves a change in the reconstruction criterion, which consists in cleaning or eliminating the noise of a partially corrupt input [19]. This is the foundation of DAEs.

In this context, the model learns a new representation from a corrupt input and is able to reconstruct the original one, without noise. This process generates a new representation of higher quality and robustness. In addition, this model, due to its operation, is less sensitive to the real noise that may exist in the input data.

At this point, it is important to emphasize that eliminating noise is not the objective of this type of model. Instead of that, the noise introduced in the first layer allows the methods to learn a higher level representation [12,18].

DAEs are models that follow this methodology for feature fusion. There are different ways to introduce noise at the input. An example could be a DAE that changes random values from the input to the value 0. The number of modified elements from the input oscillates between 30% and 50%. Even so, this amount will depend on the dimensionality of the input data and the architecture of the network. A key aspect to consider is that, when calculating the loss function, the model must compare the output obtained with the original input, without noise. As a result, the possibility of totally or partially copying the input is practically eliminated, given that the corrupt input is different from the expected output.

The DAE model is similar to a basic AE, but introduces some modifications. The objective of the DAE is to reproduce the clean input from a corrupt version of itself. Therefore, the first step is to perform a stochastic mapping of the input [39].

The input with noise is mapped through the AE to a coding through the hidden layers. Then, the output of the network is reconstructed from the coding generated in the hidden layers of the network. Throughout the training process, the network adjusts the parameters to generate an output as similar as possible to the original input. The fundamental difference is the use of a corrupt input instead of the clean one.

The goal of DAEs is to minimize the reconstruction error. There are different loss functions such as mean square error (MSE) or cross entropy. Once the error is determined, the parameters of the network are to be adjusted. There are several methods for doing this, including stochastic gradient descent (SGD) and its variants, including RMSProp and AdaGrad.

The main purpose of the different SGD methods is to adjust the parameters in such a way that the objective function is minimized. To do so, a backward propagation process goes through the network from the output layer to the input, transmitting the necessary adjustments to optimize the process. Fig. 5 presents a diagram showing the training process of a DAE.

The training process of DAE is very similar to the one used in a basic AE. Nonetheless, introducing noise in the original input makes the network more stable against noise and allows the generation of much more robust features. For this reason, the ClEnDAE model is based on this type of AE to face the task of dimensionality reduction.

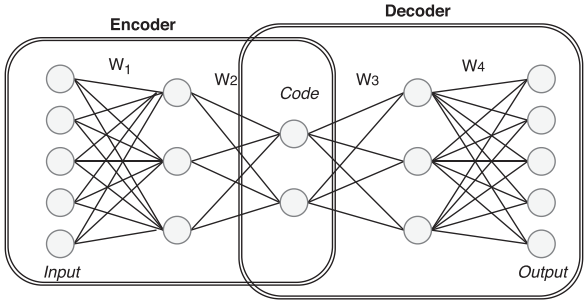

Fig. 4. AE basic architecture.

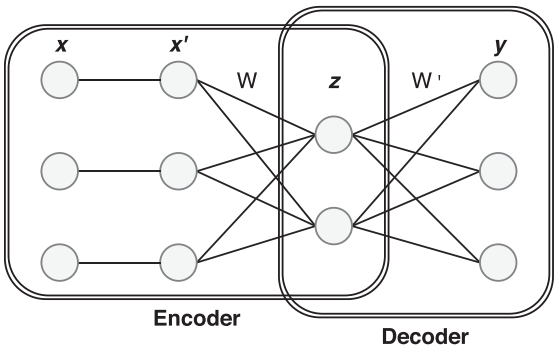

Fig. 5. DAE architecture.

2.5. Ensemble

The ClEnDAE model is based on ensembles due to their advantages and their good results in other areas [6,20]. An ensemble is a composition of several machine learning models. The main advantages provided by an ensemble-based model are:

- The information given by the different models can be combined, producing a classifier profile that cannot be obtained from a single model.
- Since each model is based on different subsets of instances and attributes obtained from the input, the ensemble is able to generalize information based on different sets, whereas a simple model commonly uses the full set and all input features.
- Ensembles can be parallelized more easily than a single model. The nature of ensembles allows parallel execution to be direct. Some traditional methods need to be redesigned to work concurrently, while the architecture of the ensembles make them perfectly usable for this type of execution.

From a general point of view, methodologies used in ensembles consist of a series of models that work on the original dataset in different ways and combine the outputs to generate a single result. There are two large groups of ensembles: sequential and parallel. Additionally, ensembles can be homogeneous, if the components used are similar, or heterogeneous if there is a difference in the components of the ensemble. Besides, there are different methodologies available to develop an ensemble, the best known and most widely used are:

- Bagging: This methodology, also known as Bootstrap aggregation, is one of the basic paradigms used for ensemble development. This algorithm is based on the statistical method of bootstrapping to jointly evaluate the execution of several models run in parallel. Bagging does not use the complete original dataset to train each of the models, but a number of input examples that are randomly selected.
- Boosting: This classical algorithm is commonly used in the development of ensemble-based methods, which are based on the reduction of bias by combining different models. Each component is added sequentially and focuses on instances misclassified by previous components. In this way, the final model is fitted more precisely to the training set, significantly improving the classification of problematic instances for individual models.
- Stacking: It is a methodology based on a sequential structure, where outputs provided by the previous models are combined and used as the input for new methods that generate a new set of predictions. It is important to note that, in this case, all the models that make up the ensemble use the complete input set.

Fig. 6 represents the different types of ensembles that exist in the literature and that have been previously described.

In conclusion, methods based on ensembles allow to expand the search space of the traditional classification algorithms, as they incorporate several components that use different sets which in turn come from the original data. The aforementioned advantages have a positive impact on the final performance of this type of algorithm. As a result, the ClEnDAE proposal made in this paper incorporates the use of this methodology.

3. ClEnDAE: A classifier based on ensembles with built-in dimensionality reduction through denoising autoencoders

After having described the main concepts of the theoretical framework associated with our proposal, this section presents ClEnDAE. It is an ensemble-based classification method that incorporates dimensionality reduction using DAEs.

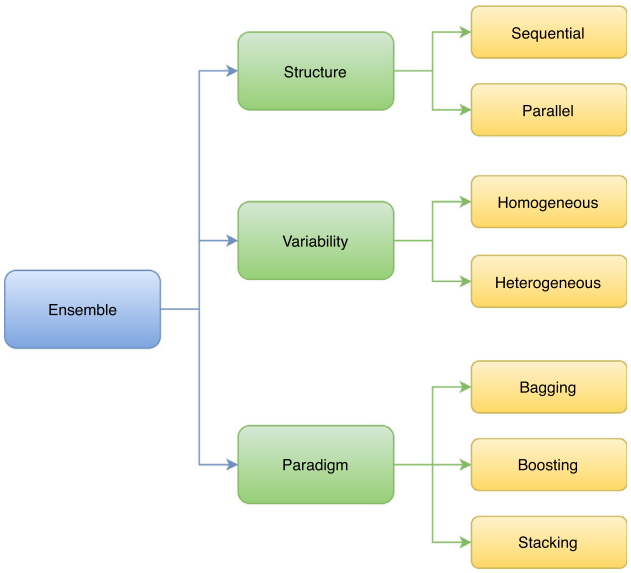

Fig. 6. Ensemble typology.

3.1. ClEnDAE foundations

This study proposes the ClEnDAE model, a method based on two fundamental pillars: DAEs and ensembles. This work is derived from the hypothesis that an ensemble-based classifier will improve its predictive performance when incorporating the use of DAEs to reduce dimensionality. This approach leverages the ability of DAEs to reduce dimensionality [19] and the advantages of using ensembles.

ClEnDAE trains each of the different components with a random subset of both instances and features obtained from the original dataset. Additionally, this method is grounded on the execution of several models in parallel with the aim of reducing variance. As a result, generated predictive results are more stable and precise. Last of all, ClEnDAE uses a voting system, one of the most common techniques to combine the outputs of the different models, meaning that the output of the ensemble corresponds to the majority result obtained by the constituent models.

The number of components of an ensemble depends on many factors, including the properties of the input data and the number of output classes. In our experiments, the same ClEnDAE configuration is applied to all the datasets that make up the experimental framework. The performance of ClEnDAE with a large set of data with different characteristics can thus be determined from a general perspective. For this reason, the number of components is 90, since it represents a significant number of units for each of the three DAE architectures being considered and an appropriate generic value for the different datasets. Nonetheless, when applying ClEnDAE to solve a specific problem, the number of components should be adjusted to data characteristics in order to obtain optimal results. Therefore, the ClEnDAE algorithm has a specific parameter that defines the amount of components in the ensemble.

Specifically, each unit contains a DAE to reduce the dimensionality of the input space. The new feature space is then used by a classifier so each component uses a different feature space generated by its DAE. The final result corresponds to the majority output provided by the 90 models. On account of this, the process followed has three fundamental phases:

- Dimensionality reduction: Each model generates a new feature space out of a subset of instances and features of the input data. In this phase, DAEs with different architectures are used. These architectures are adapted to generate a space of lower dimensionality.
- Classification: Data generated by each of the DAEs are used by the same type of classifier. ClEnDAE has four possible classifiers: kNN, SVM, MLP and C4.5. The method is selected by means of one of its parameters.
- Output: The final response of the model will be the majority output by the different classifiers.

In this context, it is important to emphasize that there are different architectures in the DAEs present in the components of ClEnDAE. More specifically, they use three architectures with a variable degree of dimensionality reduction. The structure of the ClEnDAE model, built out of 90 components, has 30 models that reduce dimensionality down to 75%, 30 more that reduce it to 50% and 30 other that reduce it to 25%. As a consequence, within each group, DAEs have a hidden coding layer where the number of neurons corresponds to 75%, 50% and 25% of the original features, respectively. The use of several architectures

allows the model to learn high-level characteristics associated with datasets with a different compression degree, increasing the amount of relevant information in comparison to configurations of a single type. The choice of these architectures is based on a series of previous works that show their better performance against other possible values [18,19].

Besides, as indicated above, each component uses a different subset of instances and features obtained from the input. To achieve this, a certain percentage of both instances and attributes are randomly selected. This allows the model's search space to be diversified. In this way, the ClEnDAE algorithm treats high dimensionality from different perspectives.

- In relation to training instances: Each model randomly selects a different set.
- From the perspective of the features: In the first place it makes a random sample with replacement of the variables and, afterwards, it learns a new space of features using the DAE.

In short, ClEnDAE uses a philosophy that is similar to Random Forest, one of the most popular algorithms based on ensembles, when training each component with a random selection of instances and features of the original set. However, ClEnDAE incorporates more phases to tackle dimensionality reduction from another perspective:

- Each component obtains a new representation of the input variables by means of a DAE out of different random subsets.
- Using new feature spaces, each component obtains a classification model.
- The final output is obtained by combining the outputs of each component by a majority vote.

3.2. Method description

ClEnDAE consists of three phases. Firstly, several random sets of instances and features are obtained from the original input. Particularly, each subset contains 63.2% of the training examples and 75% of the input features, and both cases involve a random selection with replacement. The random choice of 63.2% of the instances ensures that all examples are selected at least once during the training process, considering a sufficiently large number of components, like in this study. This percentage is common in bagging and boosting methods. Furthermore, the selection of 75% of the features is based upon a series of previous studies showing that this reduction degree produces better performance [18,19]. To sum up, this random selection of both instances and characteristics implies that the search space is diversified in both directions.

Secondly, these subsets are used to train the 90 components of the ensemble. This process consists in training a DAE to achieve a new representation of lower dimensionality and then training a classifier with the new space to obtain the predic-

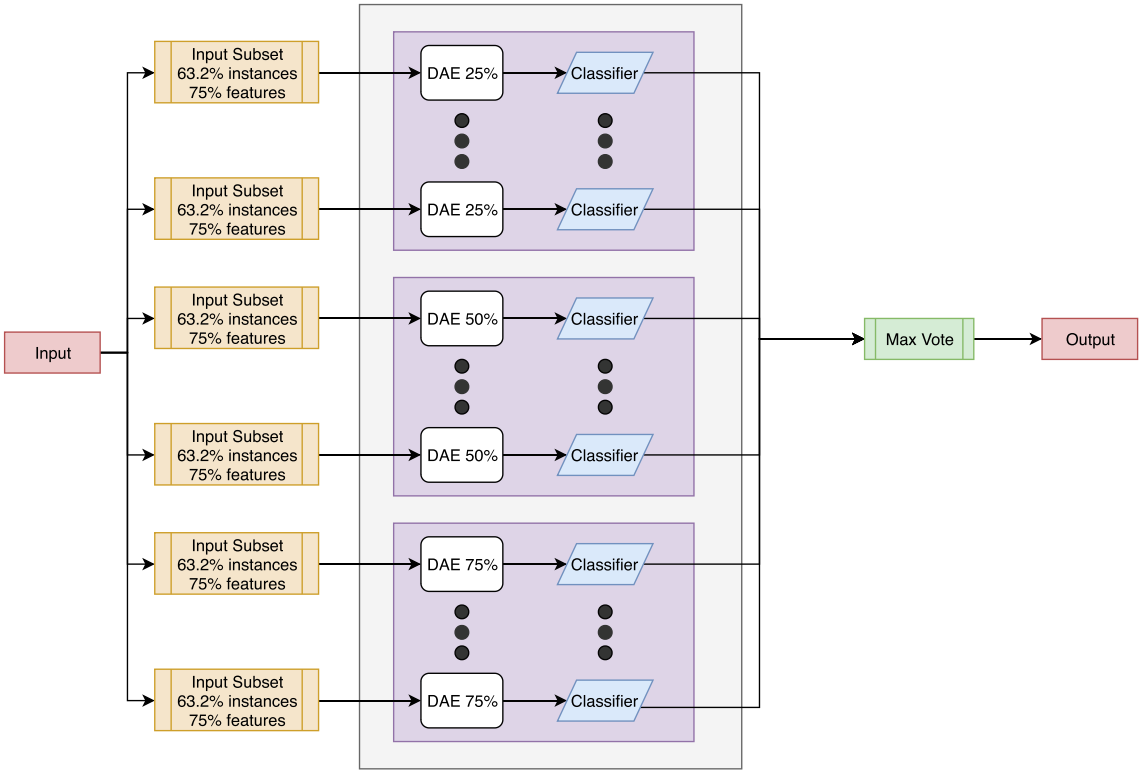

Fig. 7. Architecture of the ClEnDAE model proposed in this study.

tion. Lastly, the outputs of all components are added to generate the final output of the model. Fig. 7 displays the architecture of the ClEnDAE model proposed in this paper. The image shows all three phases: random selection, reduction of dimensionality and classification; as well as different architectures of DAEs. In addition, Algorithm 1 presents the pseudo-code of ClEnDAE.

Algorithm 1: ClEnDAE algorithm's pseudo-code.

Inputs:
TrainData ▷Train Data
TestData ▷Test Data
NumComponent ▷Number of components in ensemble
ArqDAE ▷Architecture DAE
Clas ▷Classifier
Par ▷Parameters of classifier
1: ▷Training:
2: *ensemble* ← initEnsemble(*NumComponent*, *ArqDAE*) 3: **foreach** *component* **in** *ensemble***do**
4: ▷Instances and features selection:
5: *componentData* ← randomSelection(*TrainData*) 6: ▷Train DAE:
7: *daeModel* ← trainDAE(*componentData*, *component*) 8: ▷New representation through DAE:
9: *componentData* ← applyDAE(*componentData*, *daeModel*) 10: ▷Train classifier:
11: *clasModel* ← trainClassifier(*componentData*, *Clas*, *Par*) 12: ▷Update component:
13: *ensemble* ← updateComponent(*component*, *daeModel*, *clasModel*) 14: **end for**
15:
16: ▷Classification:
17: *result* ← classification(*TestData*, *ensemble*)
18: **return** *result*
19:
20: **function**trainDAE*componentData, component*
21: *modelData* ← corruptedInput(*componentData*)
22: *daeModel* ← () 23: **foreach** *layer* **in** *component* **do**
24: *sizeLayer* ← getSizeLayer(*modelData*, *layer*)
25: *daeModel* ← addDAELayer(*sizeLayer*) 26: **end for**
27: *daeModel* ← compileDAEModel(*modelData*, *daeModel*)
28: *component* ← addDAEModel(*daeModel*)
29: **return** *component* 30: **end function**
31:
32: **function**compileDAEModel*modelData*, *daeModel*
33: **foreach** *instance* **in** *modelData* **do**
34: *outPut* ← feedForwardDAE(*daeModel*, *instance*)
35: *error* ← calculateDeviation(*instance*, *outPut*)
36: *daeModel* ← updateWeightsDAE(*daeModel*, *error*) 37: **end for**
38: **return** *daeModel* 39: **end function**
40:
41: **function**classification*TestData*,*ensemble*
42: *error* ← 0 43: **foreach** *instance* **in** *TestData***do**
44: *outputs* ← () 45: **foreach** *component* **in** *ensemble***do**
46: *newRep* ← applyDAE(*instance*, *component*)
47: *componentOutput* ← applyClassifier(*newRep*, *component*)
48: *outputs* ← addOutput(*outputs*, *componentOutput*) 49: **end for**
50: *finalOutput* ← selectMajorityOutput(*outputs*))
51: *error* ← updateError(*instance*, *outPut*)) 52: **end for**
53: *result* ← calculateError(*TestData*, *error*)
54: **return** *result* 55: **end function**

The input parameters of Algorithm 1 are described below:

- *TrainData* and *TestData*: The training and test data processed by the algorithm.
- *NumComponent*: The number of models that make up the ensemble. This value has been set to 90, as explained above.
- *ArqDAE*: The configuration of the different structures of the DAEs used in the components of the ensemble. Three different architectures have been considered in this study, so there are 30 models with each configuration. This parameter sets the number of layers and elements per layer for each configuration.
- *Clas*: The classifier used by the algorithm. For the purposes of this study, four different classifiers have been considered: kNN, SVM, MLP and C4.5. All components use the same classifier.
- *Par*: The parameters associated with the selected classification algorithm.

The operation of the method is divided into two different blocks. The first part (lines 1–14) carries out the training of the ClEnDAE model. The second part of the code (lines 16–18) performs the classification of the test set to evaluate this model.

The training process has two fundamental objectives. First of all, ClEnDAE focuses on learning a new representation of the input data to tackle dimensionality reduction using DAEs. Secondly, the selected classification model is trained. These phases are carried out in each of the ClEnDAE components (lines 3–14). The code associated with this phase is described in more detail below:

- The initialization of the ensemble is carried out in line 2. This process consists in creating the model structure and all the associated components. The *NumComponent* parameter defines the number of components that will be created at this point. Likewise, the *ArqDAE* parameter defines the different architectures that the DAEs of said components will contain, and those will also be initialized.
- Lines 3–14 correspond to a loop that performs the following tasks for each component of the ensemble:
  - Line 5: Random selection of 75% of the features and 63.2% of the instances of the training data. This allows to diversify the search space of the different components.
  - Line 7: Training of the DAE responsible for obtaining a new representation with less input data dimensionality.
  - Line 9: A new representation of the input subset is computed using the DAE trained in the previous step.
  - Line 11: Training of the selected classifier. In this study, four different classifiers have been considered: kNN, SVM, MLP and C4.5. In this sense, it is important to emphasize that not all models need this phase (for example kNN), in which case only the initialization of the classifier is performed.
  - Line 13: The component is updated with the DAE and the previously trained classifier.

Throughout the process described above, each component of ClEnDAE conducts a DAE training by using the training data to learn the weights of the network that produce a better representation of the input. This stage corresponds to the *trainDAE* function detailed below:

- The first step is to introduce noise into the input (line 21). This is key when operating DAEs, which consists in learning to rebuild the input from corrupted data.
- As can be seen in the code (for loop, lines 23–26), the structure corresponding to the DAE is created, adding all the necessary layers. Each iteration of the "for loop" generates a new layer of the model.
- Once initialized, the DAE is trained using the training data in order to update its parameters. Procedure *compileDAEModel* iteratively compiles the structure of the DAE. This function is encoded in lines 32–39. In order to carry out this process, the model generates a new representation for each training instance (line 33), computes the reconstruction error of the original input (line 34) and updates the weights to minimize the error (line 35). Last of all, the DAE model is returned using the function (line 37).
- The DAE model obtained in this function is added to the component (line 28) and this is returned (line 29).

The *classification* function (lines 41–55) carries out the classification. In this process each component obtains the class prediction for test instances (lines 45–49) and all the information is added to generate the final output, selecting the most common output (line 50). Here are the following stages required to compute this prediction in each component:

1. Firstly, a new coding of the instance is calculated using the DAE model included in each component during the training phase (lines 46). This representation generates a new feature space with a higher level but lower dimensionality.
2. Secondly, the coding computed in the previous step is used as input of the classification algorithm, generating the final prediction of the current component (line 47). Each one has an associated classification model that was produced during the training phase.
3. Thirdly, the output of each component is saved (line 48).

Lastly, the majority output is selected (line 50) and the error rate is determined, comparing the obtained output to the actual one (line 51). The global error value is returned by the algorithm to evaluate its behavior (lines 53–54).

As can be seen, the classification phase, which has been explained above, uses data of lower dimensionality than the original input, reducing the negative effects caused by the existence of too many features.

Several points regarding the previously mentioned algorithm must be clarified. For example, the DAE training process has been executed using a mini-batch gradient descent. This method is a variation of the gradient descent, which divides the dataset into small batches that are used independently to calculate the error and modify model parameters. This operation is suitable when a high-dimensional dataset is used.

As aforementioned, all the architectures have a single hidden layer, in addition to the input and output layers. Different studies in the literature have shown that the use of a single hidden layer produces better results when facing the task of dimensionality reduction [18,19]. Similarly, having several architectures implies generating different degrees of dimensionality reduction. The components will reach high level characteristics with several degrees of generalization and the variability of each of their votes will increase.

3.3. Sensitivity study on the ClEnDAE parameters

In this section, an analysis of the parameters that are used by the ClEnDAE method is carried out, allowing to justify the values denoted in the previous sections.

Specifically, two parameters will be studied: DAE compression percentage (architecture) and the number of components of the ensemble. To do this, a subset of datasets will be selected from those used in the subsequent experimentation. The final classification has been carried out using the kNN classifier.

Initially, the architecture of the internal DAEs is analyzed. For that purpose, the architectures given in Table 1 have been considered. As shown in Table 1, they have a different number of hidden layers and neurons in each layer.

In this experimentation, all components of the ensemble are deemed to have the same architecture. Thus, Table 2 displays the predictive performance of each configuration. The tables highlight the best results in bold.

These results show that hidden layer architectures generally perform better than hidden multiple layer architectures. Likewise, in all cases, the best predictive performance is obtained with the 75% configuration. However, the possibility of combining different models has not been considered. Table 3 presents a comparison between the best previous configuration and an architecture that contains different types of models (Arq 1, Arq 2 and Arq 3).

The comparison evinces how increasing the diversity of the component architecture contributes to improving predictive performance. For this reason, in this study, we have used hidden-layer architectures that have a variable reduction percentage.

Now that the architecture of the DAEs that compose the ensemble is selected, the number of components of the model is going to be analyzed. Table 4 shows the predictive performance of the ClEnDAE method, considering executions with different number of components.

The previous results expose that the best performance is obtained with 90 units. Because of this, the ClEnDAE model is made up of 90 components in the subsequent experimentation.

3.4. ClEnDAE complexity

The objective of this section is to analyze the computational complexity of the ClEnDAE algorithm. As explained above, Algorithm 1 is fundamentally divided into two parts: the training of the model and the classification of the test set. Firstly,

Table 1
Architectures used in the experimentation.

		Number of neurons (%)			
	# hidden layers	Layer 1	Layer 2	Layer 3	Architecture
Arq 1	1	25	-	-	(25)
Arq 2	1	50	-	-	(50)
Arq 3	1	75	-	-	(75)
Arq 4	3	150	25	150	(150, 25, 150)
Arq 5	3	150	50	150	(150, 50, 150)
Arq 6	3	150	75	150	(150, 75, 150)

Table 2
kNN classification results of ClEnDAE with different DAE architecture.

Dataset	Arq 1	Arq 2	Arq 3	Arq 4	Arq 5	Arq 6
coil2000	0.541	0.554	**0.558**	0.520	0.533	0.539
image	0.922	0.938	**0.944**	0.902	0.918	0.929
madelon	0.588	0.592	**0.605**	0.559	0.573	0.582

Table 3
kNN classification results of ClEnDAE with the best DAE architecture and combine different architectures.

Dataset	Arq 3	Arq 1 + Arq 2 + Arq 3
coil2000	0.558	**0.562**
image	0.944	**0.954**
madelon	0.605	**0.614**

Table 4
kNN classification results of ClEnDAE with different number of components.

Dataset	60	70	80	90	100	110
coil2000	0.549	0.555	0.559	**0.562**	0.561	0.558
image	0.947	0.945	0.949	**0.954**	0.952	0.951
madelon	0.603	0.607	0.611	0.614	**0.617**	0.615

considering N the number of training instances, n the features of the dataset, $M$the number of components of the ensemble and Clt the complexity of classifier training, computational complexity for the training block would be the following:

$$\begin{aligned} C_{ClEnDAE_T} &= O(1 + M(\tfrac{3}{4} \times n + M \times N + N + Clt + 1) = \\ &O(1 + M(C \times n + M \times N + N + Clt)) = \\ &O(1 + M \times n + M^2 \times N + M \times N + M \times Clt) = \\ &O(M \times (n + M \times N + N + Clt)) = \\ &O(M^2 \times N + M \times N + M \times n + M \times Clt) \end{aligned} \tag{1}$$

Expression 1 shows the quadratic factor M^2 as being the element of greatest weight. Nevertheless, this will be the case only when $M \geqslant N/n$. Since this work focuses on high-dimensional datasets, it is possible to assume that $M \ll N$ and $M \ll n$. Consequently, M^2 would remain as an additive factor to $M \times Clt$, as it has the greatest weight.

Secondly, considering $N$the number of instances of the test set, $M$the number of components and Clc the complexity of classifier testing, the computational complexity of the classification block corresponds to:

$$C_{ClEnDAE_C} = O(N \times M \times Clc) \tag{2}$$

To conclude, the computational complexity of both parts of the algorithm depends on the parameters of ClEnDAE. This factor also depends on the complexity of the selected classifier.

3.5. ClEnDAE contributions

ClEnDAE is an ensemble-based classifier that incorporates DAEs in order to reduce the dimensionality of the input data and mitigate the associated negative effects. The main contributions of this proposal are the following:

- ClEnDAE treats high dimensionality from different angles. Each component uses a random selection of both instances and features of the original set. Moreover, the model learns a new representation of the feature space using DAEs.
- The application of DAEs by ClEnDAE to tackle dimensionality reduction delivers improvements in the predictive performance of many traditional classifiers, as described in the experimentation.
- ClEnDAE is a model based on ensembles. It relies on the fact that the predictive performance of several components working together is greater than that of the models used independently.
- Each component of the ensemble uses a random selection of both instances and features to train. Because of this, the ClEnDAE search space is diversified, improving predictive performance.
- The algorithm is parameterized to use four different algorithms in the classification phase: kNN, SVM, MLP and C4.5. This allows to evaluate the behavior of the method considering four classic classifiers belonging to different families.

There are methods in the literature that address the problem of high-dimensional data from different perspectives. Similarities and differences between the ClEnDAE method and some of them are listed below:

- SGDNMF [40]: This method aims to extract high-level relationships from the input data to improve the dimensionality reduction task performed by the classical NMF algorithm. Like ClEnDAE, this method generates a new feature space from the input data using a deep learning model. However, whereas ClEnDAE is a classification method that incorporates a dimensionality reduction phase, the SGDNMF algorithm is a data representation method, so the purpose of both algorithms is different.

- NSSRD [41]: This algorithm addresses the task of feature selection, its main advantage is that it uses the feature graph together with the data space to improve the performance of this task. Likewise, the method incorporates other mechanisms to guarantee the quality of the selected characteristics. In the same way as ClEnDAE, this method tackles the task of dimensionality reduction. Nonetheless, each method uses a different paradigm. While ClEnDAE generates new features by merging the originals, the NSSRD method selects original features directly. The ClEnDAE method incorporates an additional classification phase, where the model output is generated.
- LDSSL [42]: This method applies the decomposition matrix to feature selection, preserving both local discriminant structure and local geometric structure of the data. Similar to the ClEnDAE method, LDSSL aims to preserve the structure of the input data and reduce the dimensionality of the input data at the same time. However, while ClEnDAE generates new characteristics based on the original data, LDSSL selects those that are most relevant. Hence, the philosophy used to deal with dimensionality reduction is different.
- DSNMF [43]: This method, like ClEnDAE, performs dimensionality reduction of the input data in two phases. However, its methodology is different. First, the DSNMF algorithm looks for a low-dimensional representation of the data. Next, it makes a selection of the most relevant characteristics. This allows a new space to be obtained with smaller yet much more representative features. DSNMF is a feature selection method, while ClEnDAE incorporates the feature fusion as an internal phase to classify the input data.

In addition to presenting the ClEnDAE model, this study intends to demonstrate how it properly mitigates the effects of high dimensionality when addressing classification tasks. An exhaustive experimental analysis is performed to this effect in Section 4. This study is divided into two phases: a comparison of the predictive performance of the model using different traditional classifiers, and an analysis of the results obtained by means of the ClEnDAE model with respect to other traditional dimensionality reduction algorithms.

4. Experimental Study

There are pre-exisiting studies that have shown both the benefits of using AEs to deal with the feature fusion task [12,18] and some that focus on the use of DAEs [19]. The main objective of the experimentation developed in this paper is to demonstrate the improvements obtained when treating this task using ClEnDAE. To that end, it is necessary to conduct an exhaustive comparison between a basic DAE model, a model based on the ensemble of DAEs and the results obtained from the original data. The process to execute each model is as follows:

- Basic classifier (baseline): In this case, the original raw data is used to classify by means of algorithms kNN, SVM, MLP and C4.5.
- Basic DAE: The first stage involves carrying out the task of feature fusion. As a result, different lower-dimensional subsets are obtained for each original dataset. The architecture used has a hidden layer that compresses 75% of the original characteristics. This configuration has shown the best performance in previous studies [3,18]. Then, classification is performed with different traditional methods. Each algorithm uses the subsets generated by the basic DAE. The classifiers applied in this phase are: kNN, MLP, SVM and C4.5.
- ClEnDAE: The algorithm proposed in this study is completed with four different classification algorithms: kNN, SVM, MLP and C4.5. Four executions are made for each dataset, changing the parameter corresponding to the classifier.

Afterwards, the classification performance of the different methods can be compared, in order to establish which model offers the best results.

The following subsections present the development of the experimentation explained above. Essentially, it intends to achieve the following goals:

- To determine the performance of our proposal, ClEnDAE. To this end, an exhaustive comparison is performed between the results from this model and the results from a basic DAE model and the original data, as detailed in SubSection 4.2. This comparison is made for each dataset and for each of the four classification algorithms considered in this study.
- To compare the results obtained with the ClEnDAE model and the four traditional methods of dimensionality reduction: LDA, PCA, ISOMAP and LLE, as shown in SubSection 4.4.
- To present some general conclusions on the results obtained in this study in SubSection 4.3 with the following aims:
  - To analyze the performance of the different models according to each classifier.
  - To identify which model offers the best performance for each case.
  - To suggest new lines of future research based on the outcomes of the study.

Before presenting the results of the experimentation, SubSection 4.1 describes the framework used in this study: datasets, metric, statistical tests, AEs architecture and classification algorithms.

4.1. Experimental framework

To establish a comparison between the models considered in this study, it is necessary to use a broad set of datasets with varied characteristics. Table 5 presents the datasets included in this experimentation, as well as their most relevant traits. The **Field** column presents the field of application.

Additionally, the area under the ROC curve (AUC) is employed to evaluate the different models in this study. The main advantage of this metric is that it offers a robust view of the predictive performance of the analyzed methods. In other words, AUC provides a global view of the results, while other metrics such as Accuracy or Precision give a more partial view [18]. AUC is the probability that a model will classify a randomly chosen positive instance higher than a randomly chosen negative one. Eq. (3) shows the definition of AUC:

$$AUC = \int_{\infty}^{-\infty} TPR(T)FPR(T)dT \tag{3}$$

where *TPR* is the true positive rate and *FPR* is the false positive rate.

This experimentation has involved the AUC metric for evaluation of both binary and multi-class datasets. Package *pROC* for R has been used for this purpose. This framework contains a set of tools to visualize and analyze the ROC curves. More precisely, the package documentation indicates that the AUC calculation for multi-class datasets is performed as defined by Hand and Till in [44].

Once the method for evaluating the results obtained has been established, it is essential to present the statistical tests applied in this study. These tests analyze the results to see if they are statistically significant and, consequently, different conclusions can be reached. The tests used are as follows:

- The Friedman test [45] allows to obtain a ranking of the different models. In this manner, it is possible to establish which model has a better performance from a general point of view.
- The Li post hoc test [46] for the Friedman test is a non-parametric method. This test is applied to identify significant differences between the models compared in this experimentation. As a means to do so, it compares all the models with each other. The Li test is used in this study because it is one of the best alternatives for finding significant differences when the number of samples is not very large.

Lastly, it is necessary to state the hardware equipment on which the experimentation has been conducted. The objective is to make the experimentation as reproducible as possible. The experiments were performed in a cluster composed of 8 computers, with 2 CPUs (2.33 GHz) and 7 GB RAM each. Moreover, it is important to indicate that the models based on AEs (basic and ensemble) have been implemented in the Python language, using the Keras library. Classification methods, which have been developed using the R language, are detailed in SubSection 4.1.1.

4.1.1. Classification Algorithm Framework

In this section, the data generated by ClEnDAE will be used by different classification algorithms in order to verify the performance of the feature fusion. As a result, the experimentation will provide useful information about the behavior of

Table 5
Characteristics of the datasets used in the experimentation.

Number of Dataset	Samples	Features	Classes	Type	Field
arcene	900	10000	2	Real	Medical
batch	13910	128	6	Real	Chemical
coil2000	9822	85	2	Integer	Social
dota	102944	116	2	Real	Game
drive	58509	48	11	Real	Motor
facial	2964	301	2	Real	Image
fashionmnist	70000	784	10	Integer	Image
gisette	13500	5000	2	Integer	Image
hapt	10929	561	12	Real	Activity
image	2310	19	7	Real	Image
isolet	7797	617	26	Real	Image
letter	20000	16	26	Integer	Image
madelon	2000	500	2	Real	Artificial
mfeat	2000	649	10	Real	Image
microv1	360	1300	10	Real	Biology
microv2	571	1300	20	Real	Biology
mnist	70000	784	10	Integer	Image
musk	6598	168	2	Integer	Physical
nomao	1970	118	2	Real	Technology
semeion	1593	256	10	Integer	Image

the ensemble model according to data characteristics and the classification algorithm. This subsection focuses on introducing the classification algorithms considered in this study, specifying their implementation and their main parameters.

Each classification algorithm has been obtained from a specific R package. Then, the details for each algorithm are shown, including the parameterization used, which is a fundamental aspect for the reproducibility of the experiment.

- kNN algorithm: The *kknn* package incorporates the kNN algorithm employed in this study. Furthermore, it is convenient to remark that the number of neighbors (parameter k) was 5, since it is the recommended value in the literature [18].
- SVM algorithm: Package *e1071* has provided the SVM method. In this case, the algorithm has been executed with the default parameters that can be consulted in the package documentation [47].
- MLP algorithm: The execution of MLP has been performed using packages *caret* and *RSNNS*. The parameters of this method are the default values specified in the package documentation [48].
- C4.5 algorithm: Package *RWeka* provides method C4.5. The parameters used in the execution of the method are the default values specified in the documentation [49].

It is possible to see that the default values are used in all cases. The main reason is that this study focuses on verifying the performance of an ensemble model of DAEs over a set of traditional classification algorithms, without adjusting the parameters for each of the specific cases.

4.2. Classification Algorithms Analysis

The objective of this section is to analyze the performance of ClEnDAE with different classification methods. ClEnDAE has a parameter that allows the user to select the desired classifier from four possibilities: kNN, SVM, MLP and C4.5. This makes it possible to study how feature fusion with the ClEnDAE model infuences the predictive performance of these algorithms.

In this context, SubSections 4.2.1, 4.2.2, 4.2.3 and 4.2.4 describe the results achieved in the experimentation for algorithms kNN, SVM, MLP and C4.5, respectively. The structure of all four subsections is similar. First, a figure displays the classification results (AUC) generated with the ClEnDAE model, the basic DAE and the raw data. This is accompanied by a ranking of the different models and the results of the Li test, in order to determine whether the results are statically significant.

4.2.1. kNN

The classification results for the kNN algorithm are showcased in Fig. 8. This image displays the results obtained with the ClEnDAE model, the basic DAE and the original data for each dataset. This visual representation helps observe that the ClEnDAE model generates the best predictive performance in most cases. In particular, in 17 out of 20 cases the best performance is obtained with the ClEnDAE model and in 2 out of 20 cases it is obtained with the raw data. It is important to note that the ClEnDAE method performs better than the basic DAE model in all cases except for one, in which they tie. In summary, in 85% of the cases the ClEnDAE model improves the results from a basic DAE model and the plain data.

These results determine that a clear improvement in predictive performance is obtained when the dimensionality is reduced using the ClEnDAE model. A previous study shows how DAEs can improve their predictive performance with distance-based algorithms, specifically, kNN [19]. These methods based on distances between examples are affected by the high dimensionality of the input data, since in this context the distances are much less significant. This has a direct effect on the predictive performance achieved by the IBL algorithms. Incorporating DAEs to reduce the dimensionality with feature fusion makes distances more significant. Likewise, the new features group the most relevant information of the input space. As a result, there is an improvement in the predictive performance of the classifier.

In this sense, the ClEnDAE model, which internally trains a large number of basic DAE models, improves the results generated by an individual DAE. The main reason is that each of the DAEs that compose the ensemble can learn certain features of the data and, thus, generalize a larger amount of relevant information than an individual model.

Finally, Table 10 included in A.1 contains the specific results of the experimentation described above. The best results are obtained with the ClEnDAE method in most cases. This table confirms the conclusions reached from the previously analyzed figure.

4.2.2. SVM

In this subsection, classification results for the SVM algorithm are presented in Fig. 9. The general trend followed by the kNN algorithm is maintained in this case. Overall, the predictive performance of the SVM algorithm improves by using the data synthesized by the ClEnDAE model. Reviewing the data in detail, the ClEnDAE model obtains the best results in 16 out of 20 cases and the model that uses the original data works best in 2 out of 20 cases. There is one dataset (*coil2000*) for which the three models generate similar results and another dataset (*mfeat*) for which the models that use DAE (ensemble and basic) have the same performance. Similarly to the kNN case, these data reveal that the ClEnDAE algorithm performs better with all datasets as compared to the basic DAE model, except in two cases where they tie. In short, the ClEnDAE model produces the best results in 80% of the cases (not counting the ties).

The predictive performance of the SVM algorithm when using ClEnDAE significantly improves as compared to the basic model and the original data. SVM is based on the maximization of the distances between instances of different classes [32].

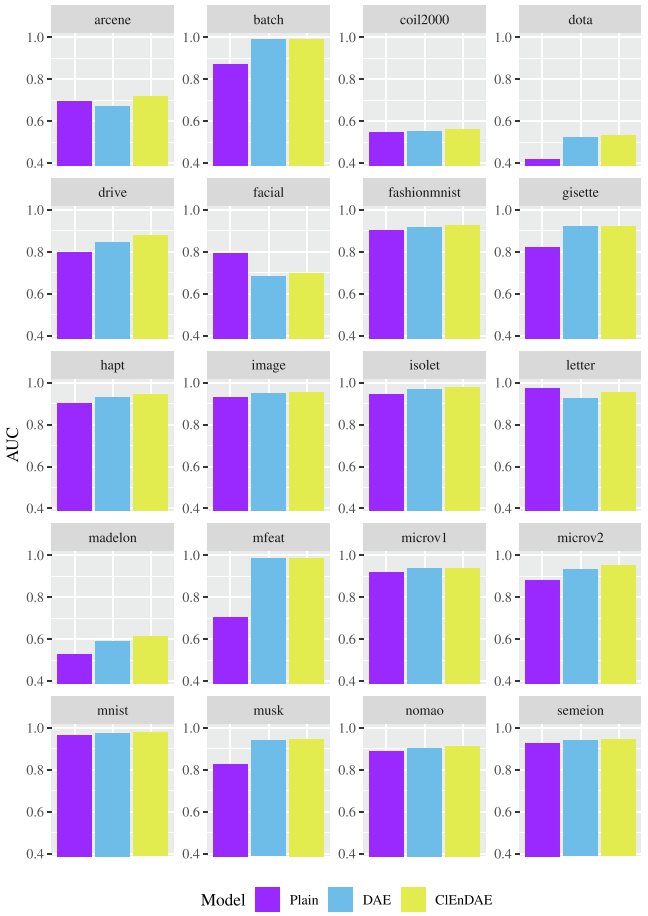

Fig. 8. AUC results for kNN algorithm.

However, these distances are equalized in spaces of high dimensionality. This implies loss of performance in this type of model and the need to apply methods that reduce the input space to mitigate the effects [5].

In this study, the proposal based on DAEs carries out a feature fusion that reduces the input space, making the distances more significant. In the same way, the model based on ensembles allows several DAEs to be trained simultaneously, which can generate different feature spaces that prevent it from discarding relevant information. Consequently, the graphs shown in this subsection confirm that the proposed model generates improvements in the behavior of the SVM algorithm.

Table 11 (see A.2) displays the results of the different models after applying the SVM algorithm. These data verify that the ClEnDAE model produces the best performance for most of the datasets used.

4.2.3. MLP

Fig. 10 represents the predictive performance of the different models. These data confirm the trend marked by the previous algorithms. The best results are obtained using the data that is generated by the ClEnDAE model, namely, feature fusion which was made with the proposed model has a positive effect on the MLP performance. A more detailed analysis of the results shows that the ClEnDAE model reaches the best performance in 16 out of 20 cases, the model based on the plain data works best in 2 out of 20 cases and the basic DAE model does not generate the best results in any case. In addition, there is a dataset (*coil2000*) for which all models provide similar outcomes and another dataset (*musk*) in which DAE-based models (basic and ensemble) produce the same value. This algorithm confirms the aforementioned tendency: the ClEnDAE model generates better results than the proposal based on basic DAE in all cases, with the exception of two ties. In short, the ClEnDAE model obtains the best predictive performance in 80% of the analyzed cases (excluding ties).

At this point, it is possible to state that the predictive performance of the MLP algorithm improves when using the ClEnDAE model to reduce the dimensionality of the original feature space. The MLP algorithm is a basic neural network whose fundamental objective is to classify a series of instances based on training examples [25]. The training process consists in training the network to progressively reduce the prediction error in the training data. Nonetheless, in high dimensional

Fig. 9. AUC results for SVM algorithm.

spaces, the information of each class that the network learns is less meaningful, therefore, the predictive performance is negatively affected.

The ClEnDAE model, proposed to reduce the input space, is fundamentally based on ANNs. Because of that, the fusion of previously generated features can be seen as a series of previous layers that generate the input of another network. In the literature, there are classification models that incorporate the DL-based feature fusion process internally, although this phase is similar to the one performed in this study [14]. In both cases, the dimensionality reduction phase allows the generation of new features that group relevant information from the input data. As such, the functioning of the classification algorithm MLP improves considerably.

In order to visualize the aforementioned outcomes more minutely, Table 12 is presented in A.2. The data represent the predictive performance of the different models using the MLP algorithm for each dataset. This confirms the fact which has been described above, the ClEnDAE model generates the best results in most cases.

4.2.4. C4.5

Fig. 11 illustrates the predictive performance achieved with the C4.5 algorithm. In general, the behavior of the C4.5 algorithm is very similar to that shown by the previous algorithms and confirms the trend they have set. The classification made with C4.5 using the data generated by the ClEnDAE model performs best in most of the datasets in the analysis. More concretely, ClEnDAE obtains the best results in 18 out of 20 cases, the model based on the plain data in 2 out of 20 and the basic DAE model does not produce the best performance in any case. As a result, this analysis confirms that the ClEnDAE model improves the basic DAE for every case considered. In summary, ClEnDAE provides the best performance in 90% of cases (excluding ties).

As explained above, the performance of the C4.5 algorithm is clearly improved when feature fusion is applied through the ClEnDAE model. This fact allows to infer that the generation of new high-level features benefits tree-based algorithms. Particularly, the C4.5 algorithm follows a training process where it analyzes the attributes that cause a separation of instances in

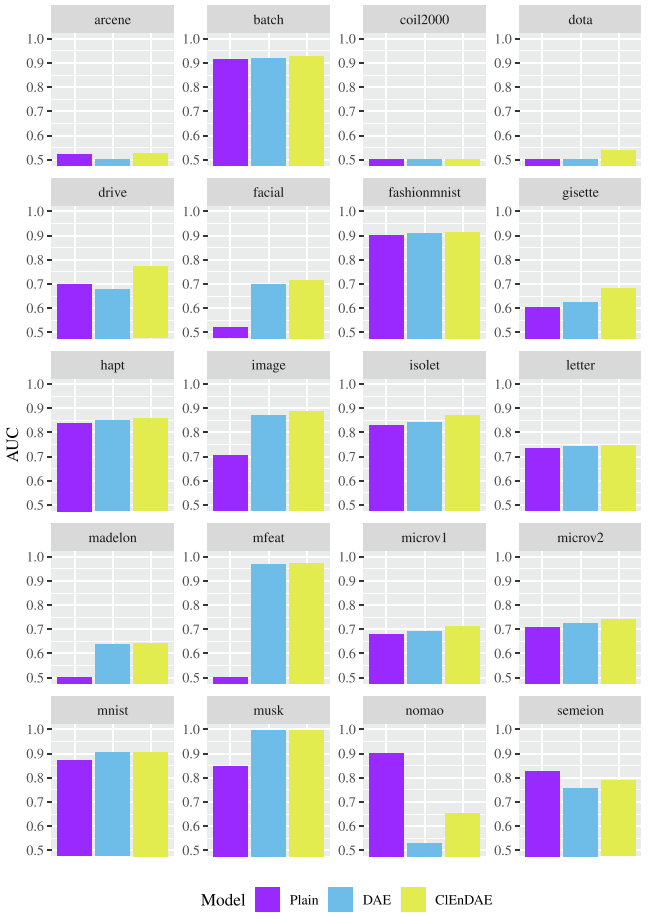

Fig. 10. AUC results for MLP algorithm.

classes [26]. This is the reason why high dimensionality of the data complicates the process of selecting the most significant features. In this circumstance, the reduction of dimensionality with ClEnDAE helps mitigate these effects.

The proposed model generates new features that group the most relevant information of the input space. As a consequence, algorithm C4.5 has a smaller and much more significant input space, enabling it to look more effectively for the attributes that divide the instances between the classes.

Finally, Table 13 (see A.1) confirms the aforementioned statements. Different data show that the best results are obtained with ClEnDAE for most of the analyzed datasets.

4.3. Results Analysis

The previous subsections have presented results from four traditional classifiers with data generated with the ClEnDAE dimensionality reduction model. Nevertheless, it is necessary to verify if these results are really significant. With this in mind, a variety of statistical tests will be used. These determine if there are actual differences between the models. The fact that these differences exist will support the results and the conclusions reached.

The main objective of this phase is to corroborate if there are significant differences between the results generated with the proposed ClEnDAE model and those provided by the basic DAE and the plain dataset. The data seen in the SubSections 4.2.1, 4.2.2, 4.2.3, 4.2.4 intuitively show that the ClEnDAE model works best, now it is necessary to determine if this improvement can be supported statistically. In this sense, the first step is to apply the Friedman test [45]. It allows to compute an average ranking for each classifier according to the results from the different analyzed models.

Table 6 presents the resulting rankings for the four classification algorithms: kNN, SVM, MLP and C4.5. In all cases, the ClEnDAE model offers the best average performance, since it is the model with the best rank among the four algorithms. The model built on basic DAE is always in second place. Finally, the model that uses the plain data ranks last overall, that is, the results generated using the original data are the lowest in general. The rankings produced by the Friedman test sup-

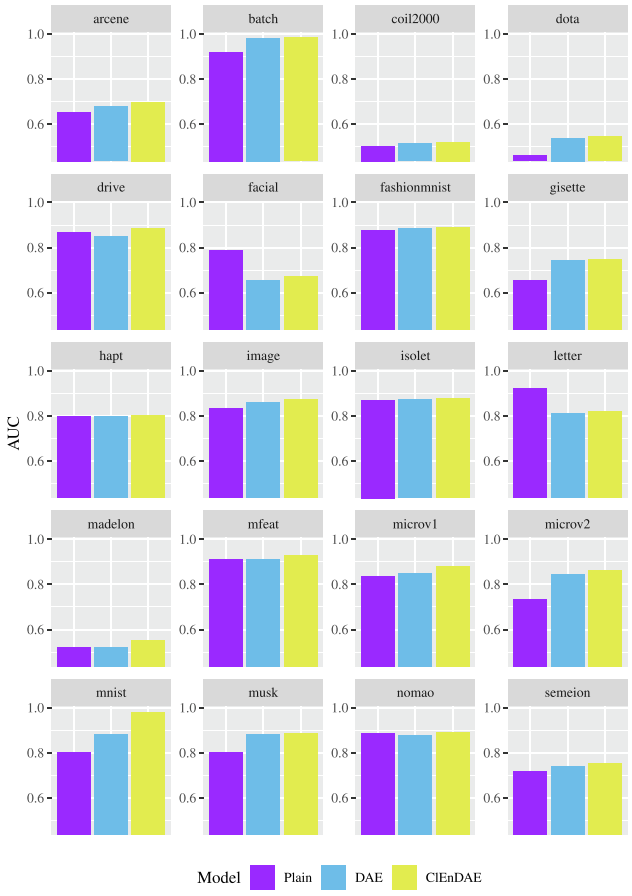

Fig. 11. AUC results for C4.5 algorithm.

Table 6
Average rankings of the different models (DAE, ClEnDAE and plain data) according to classification method.

kNN		SVM		MLP		C4.5	
AE Model	Ranking	AE Model	Ranking	AE Model	Ranking	AE Model	Ranking
ClEnDAE	1.150	ClEnDAE	1.150	ClEnDAE	1.150	ClEnDAE	1.100
DAE	2.100	DAE	2.150	DAE	2.200	DAE	2.200
Plain	2.750	Plain	2.700	Plain	2.650	Plain	2.700

port the conclusions reached in the previous subsections, which indicated that the ClEnDAE model reached the best predictive performance in most cases. Besides, this statistical test confirms that there are significant differences.

Once the Friedman test has been applied, a second test must be used to confirm whether the differences between the models observed in the rankings are significant or not. The Li test [46] is applied for this purpose. This is a specific post hoc test for the Friedman test that compares the different models present in the ranking. When used, the test offers a series of *p-values* that determine if there are differences between the models.

The data obtained after applying the Li test for the four classification algorithms can be seen in Table 7–9. It shows that there are significant differences (values in bold) between ClEnDAE and the compared models in all cases. In general, all *p-values* are very low. Particularly, the ClEnDAE model always shows significant differences to the rest of the models, which confirms the improvements in performance discussed above. Regarding the disparities between the model with plain data and the basic DAE model, significant differences are also found for the four analyzed algorithms. This corroborates that the use of DAE for feature fusion improves the performance of the classifiers, as has been described in previous studies [19].

In addition to this, Fig. 12 represents the critical distances for the different models considered. This type of graph illustrates whether there are significant differences between the different models. Specifically, if there are no significant

Table 7
Li post hoc Friedman test for classification algorithms by AE Model.

		Plain	DAE	ClEnDAE
kNN	Plain	-	-	-
	DAE	**3.983E-02**	-	-
	ClEnDAE	**4.375E-07**	**2.276E-03**	-
SVM	Plain	-	-	-
	DAE	**8.199E-02**	-	-
	ClEnDAE	**1.036E-06**	**1.702E-03**	-
MLP	Plain	-	-	-
	DAE	1.547E-01	-	-
	ClEnDAE	**2.486E-04**	**1.062E-03**	-
C4.5	Plain		-	-
	DAE	1.138E-01		-
	ClEnDAE	**4.740E-07**	**5.687E-04**	

Table 8
Average rankings considering ClEnDAE, PCA, LDA, ISOMAP and LLE by classification method.

kNN		**SVM**		**MLP**		**C4.5**	
Architecture	Ranking	Architecture	Ranking	Architecture	Ranking	Architecture	Ranking
ClEnDAE	1.275	ClEnDAE	1.475	ClEnDAE	1.875	ClEnDAE	1.750
LDA	2.825	LLE	3.000	ISOMAP	3.000	LDA	2.700
LLE	3.150	LDA	3.025	LDA	3.100	LLE	3.150
ISOMAP	3.250	ISOMAP	3.125	LLE	3.375	ISOMAP	3.350
PCA	4.500	PCA	4.375	PCA	3.650	PCA	4.050

Table 9
Li post hoc Friedman test for dimensionality reduction methods by classification algorithm.

		ClEnDAE	PCA	LDA	ISOMAP	LLE
kNN	ClEnDAE	-	**1.118E-09**	**3.867E-03**	**3.907E-04**	**5.893E-04**
	PCA	-	-	-	-	-
	LDA	-	**2.019E-03**	-	4.668E-01	5.532E-01
	ISOMAP	-	**1.769E-02**	-	-	-
	LLE	-	**1.153E-02**	-	8.415E-01	-
SVM	ClEnDAE	-	**6.6631E-08**	**6.436E-03**	**4.824E-03**	**6.436E-03**
	PCA	-	-	-	-	-
	LDA	-	**1.188E-02**	-	8.708E-01	-
	ISOMAP	-	**1.769E-02**	-	-	-
	LLE	-	**1.188E-02**	9.691E-01	8.685E-01	-
MLP	ClEnDAE	-	**3.846E-03**	**4.683E-02**	**6.001E-02**	**1.343E-02**
	PCA	-	-	-	-	-
	LDA	-	4.099E-01	-	-	6.642E-01
	ISOMAP	-	3.497E-01	8.415E-01	-	5.779E-01
	LLE	-	6.642E-01	-	-	-
C4.5	ClEnDAE	-	**4.225E-05**	**1.115E-01**	**6.852E-03**	**1.693E-02**
	PCA	-	-	-	-	-
	LDA	-	**1.724E-02**	-	2.358E-01	3.995E-01
	ISOMAP	-	2.225E-01	-	-	-
	LLE	-	**1.169E-01**	-	6.892E-01	-

differences between two models, they are connected by a horizontal line in the graph. In Fig. 12, it is possible to see the significant differences between ClEnDAE and the rest of the methods used for all classifiers.

In conclusion, the statistical tests conducted in this Subsection confirm the statements made in SubSections 4.2.1, 4.2.2, 4.2.3, 4.2.4. The ClEnDAE model provides a clear improvement in the predictive performance of the four classification algorithms analyzed in this study: kNN, SVM, MLP and C4.5. In a similar way, the basic DAE model also produces better results than the model based on plain data. Nonetheless, the proposed ClEnDAE model performs better across all classification algorithms.

4.4. ClEnDAE vs classical feature extraction techniques

ClEnDAE is a feature fusion algorithm that aims to mitigate the negative effects of the high dimensionality of the data. In previous sections, the effectiveness of the model proposed in this work has been compared with plain data and another DAE-

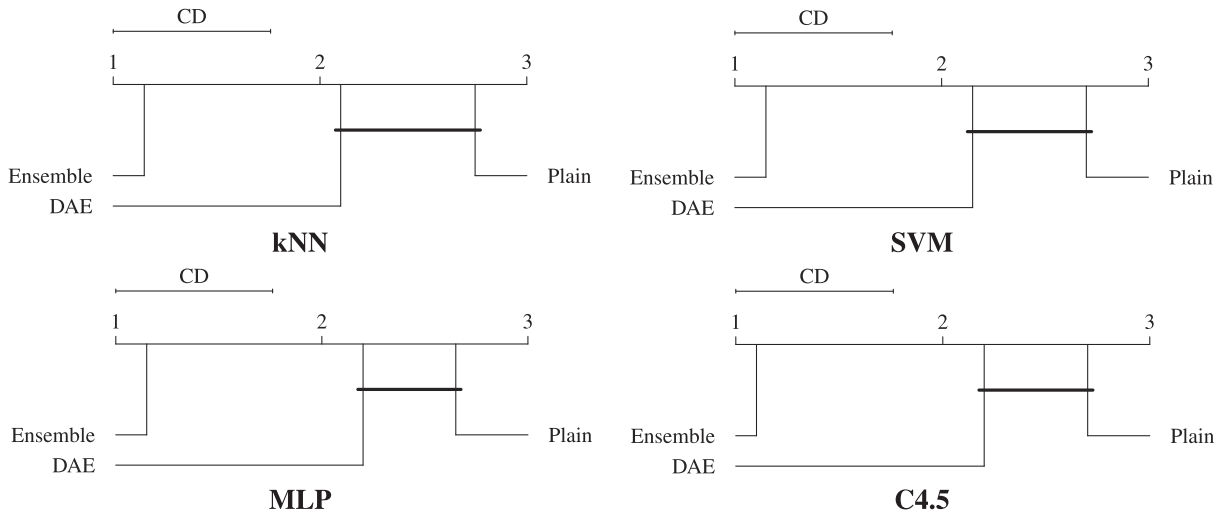

Fig. 12. Critical distance between ClEnDAE, DAE and plain models for kNN, SVM, MLP and C4.5.

based model. However, it is necessary to establish a comparison of the ClEnDAE model with other traditional dimensionality reduction methods. PCA, LDA, ISOMAP and LLE have been selected for this purpose, since they are four of the best known algorithms to treat high dimensionality of the data. To conduct the comparison, the ClEnDAE, PCA, LDA, ISOMAP and LLE methods have been used on the same datasets. The number of features generated by these algorithms will be the same in every case: 75% of the number of original features. It is essential that this value is shared in order to establish a reliable comparison and draw appropriate conclusions. Lastly, the data generated by the ClEnDAE, PCA, LDA, ISOMAP and LLE methods will be used to classify with the algorithms: kNN, SVM, MLP and C4.5. The predictive performance obtained with the different subsets provides the possibility to determine which model generates the best space of reduced dimensionality.

Classification results for the different datasets are presented in Figs. 13–16. There is an image for each classifier: kNN, SVM, MLP and C4.5, respectively. Moreover, each classifier represents the predictive performance attained from the different models: the proposal of this study, ClEnDAE, and the traditional dimensionality reduction approaches, PCA, LDA, ISOMAP and LLE. In every case, there is a bar chart for each dataset.

The observation of the results presented in Figs. 13–16 leads us to make the following observations:

- kNN: The best results for 16 out of 20 datasets are obtained with the ClEnDAE model. The LDA method only reaches the best performance in one case (*musk*), the PCA algorithm in none, ISOMAP in one case (*microv1*) and LLE in another dataset (*arcene*). There is a case in which both ClEnDAE and LLE generate the best performance. This means that ClEnDAE-based representations work best in 80% of the analyzed cases (not counting the ties).
- SVM: The ClEnDAE method achieves the best predictive performance in 16 out of 20 cases, the LLE algorithm in 3 out of 20 (it tied with ISOMAP in the *microv1* case) and PCA and LDA do not provide the best performance in any case. There is a dataset (*coil2000*) in which models ClEnDAE, LDA, ISOMAP and LLE all produce the same result. In other words, the proposed ClEnDAE model generates better performance in 80% of cases (excluding ties).
- MLP: The ClEnDAE algorithm works best in 13 out of 20 cases. The LDA method attains better performance in 2 out of 20 datasets and PCA, ISOMAP and LLE in one case each. For one of the datasets (*coil2000*) the results of the three methods are equivalent and for another (*microv1*) the performance of LLE and ISOMAP is equivalent. The model proposed in this study produces the best results in 65% of cases (without counting ties).
- C4.5: The ClEnDAE model achieves the best predictive performance in 16 out of 20 cases. The LDA algorithm works better in 3 out of 20 cases and ISOMAP in another dataset. Lastly, PCA and LLE models do not generate the best results in any case. This means that the ClEnDAE method gets better results than the rest in 80% of cases.

The results described above mark a common trend for all the analyzed classification algorithms: the ClEnDAE model works better than PCA, LDA, ISOMAP and LLE, in general. This means that the feature fusion performed with ClEnDAE builds a new feature space that provides more relevant and useful information than four of the most used models in dimensionality reduction. This pattern can be confirmed in Tables 10–17 corresponding to the algorithms kNN, SVM, MLP and C4.5, respectively. The tables highlight the best results in bold, so it is possible to infer that, as a rule, ClEnDAE produces the best performance.

Nevertheless, the previous results must be duly supported by statistical tests that establish a solid basis for the conclusions reached. To do so, it is necessary to determine if there are significant differences between the studied proposals.

Fig. 13. AUC results for kNN algorithm.

The first step consists in applying the Friedman test [45]. Different rankings for each classification algorithm are presented in Table 8. In this manner, a vision of the global operation of the different dimensionality reduction methods can be visually obtained.

The four rankings provided in Table 8 corroborate the above conclusions. The ClEnDAE model generates the best predictive performance for all classification algorithms. The LDA, LLE, ISOMAP and PCA methods perform worse than the ClEnDAE method in all the analyzed cases, with PCA producing the worst results. Moreover, LLE, LDA and ISOMAP algorithms generally have an equivalent performance. This is the first step to confirm that the ClEnDAE model offers a significant improvement as compared to PCA, LDA, ISOMAP and LLE.

The second step is to apply the Li post hoc tests [46] for the Friedman test. It compares the different methods to establish whether there are significant differences between them.

In this sense, Table 9 shows the results obtained after applying the Li test. This table represents, for each pair of models, the *p-value* associated with its statistical difference. It can be observed that the ClEnDAE model shows significant differences with respect to the LDA, PCA, ISOMAP and LLE methods in the four analyzed classification algorithms, since the *p-values* are notably low. This means that there is a statistical confirmation of the improvement provided by the ClEnDAE method with regard to PCA, LDA, ISOMAP and LLE. This supports the statements that have been made after presenting the experimental data.

Finally, Fig. 17, representing the critical distances for the different considered models, shows that ClEnDAE presents significant differences with respect to all of them.

In summary, the objective of this section is to establish a comparison between our proposal, ClEnDAE, and four traditional dimensionality reduction algorithms: PCA, LDA, ISOMAP and LLE. The results of the experimentation have shown that the proposed method typically works better than the four traditional models. These data have been adequately supported by several statistical tests that find significant differences between them. In conclusion, the predictive performance of the four classifiers used with the ClEnDAE method is much higher. This suggests that the feature fusion process performed by our proposal generates higher quality and more relevant characteristics than the other traditional proposals. This experimenta-

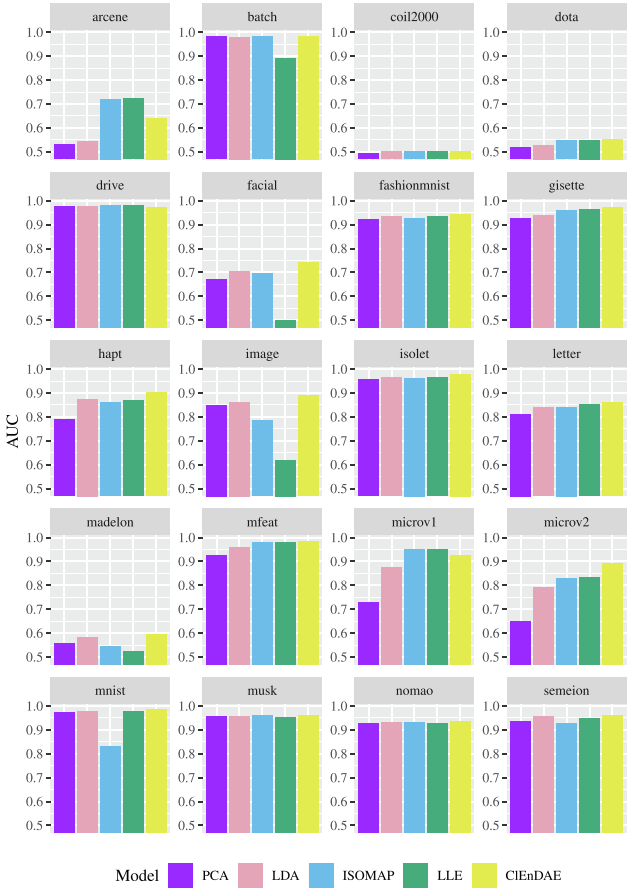

Fig. 14. AUC results for SVM algorithm.

tion provides a basis for which the ClEnDAE model could be considered an option to take into account to mitigate the effects of high dimensionality.

5. Concluding remarks

In this study, a model to deal with the task of dimensionality reduction has been proposed. This task is a challenge in machine learning, due to the negative effects that high dimensionality of the data produces in many circumstances. Specifically, this study is based on the effects of high dimensional data in different classification methodologies.

In order to address it, the ClEnDAE algorithm has been proposed. The fundamental objective of this method is to mitigate the effects of high dimensionality of the data. Thus, ClEnDAE carries out a type of feature fusion that reduces the dimensionality of the input space. The new features generated by our method add the most relevant information obtained from the original data and discard redundant or meaningless information. Good results achieved by AEs when dealing with the task of feature fusion were the inspiration which led to the development of this algorithm [12,18], in particular, some studies that show models based on DAEs performing better than traditional models [19]. The ClEnDAE method is based on two fundamental pillars: DAEs and ensembles. In this sense, the use of DAEs to address the reduction of dimensionality joins the advantages of the ensembles. Broadly speaking, a model based on ensembles allows several basic models to be trained simultaneously with partitions of the input data. This helps each model generalize relevant information of the different partitions of the data. Once the information generated by the different models is merged, the result has more relevance and usefulness than when using a single basic model.

An exhaustive study has been developed in order to determine the effectiveness of the ClEnDAE model. Fundamentally, the experimentation consists in analyzing the predictive performance of four traditional classifiers: kNN, SVM, MLP and C4.5, after performing feature and information fusion with the proposed model. This performance is compared with that obtained after using a basic DAE model and after using original data. Furthermore, a comparison with four traditional dimensionality reduction algorithms (PCA, LDA, ISOMAP and LLE) is conducted.

Fig. 15. AUC results for MLP algorithm.

The first part of the experimentation has shown that the ClEnDAE model leads to better predictive performance than the basic DAE model and the plain data model for the four analyzed classifiers. The ClEnDAE proposal obtains the best results in more than 72% of the cases. Similarly, these conclusions have been duly confirmed by several statistical tests.

Later, the second part of the experimentation has analyzed the performance of ClEnDAE against PCA, LDA, ISOMAP and LLE. This part has shown that ClEnDAE works better than four of the most common algorithms for the task of dimensionality reduction. Concretely, the ClEnDAE model obtains the best results in more than 80% of the cases for kNN, SVM and C4.5 methods and more than 65% for MLP. Statistical tests have confirmed the differences between the different models as well.

Finally, this experimentation has shown that the ClEnDAE model is able to mitigate the effects caused by the high dimensionality of the data, improving the predictive performance of different classifiers corresponding to traditional methodologies. Likewise, the provided performance is better than some of the most commonly used traditional algorithms: LDA, PCA, ISOMAP and LLE.

In conclusion, ClEnDAE is a method of feature fusion based on DAEs that deals with the problem of high dimensionality. This paper describes the developed method and an exhaustive experimentation to verify its proper operation. The provided conclusions open new lines of future work where other models of AEs, for instance, robust or contractive AEs [12], are used to generate models that are similar to ClEnDAE. The ClEnDAE method could also be applied to solve specific problems, adapting its structure and functioning to make it more effective.

The proposed algorithm can be improved by adjusting its parameters to the specific input data or by introducing new models for both classification and dimensionality reduction. Moreover, the ClEnDAE method can be related to emerging lines of research such as data lake [50]. In this regard, the ClEnDAE method can take the raw data stored in this type of system, perform the learning and feature fusion processing and, last of all, store the generated data in another data lake for later use.

Fig. 16. AUC results for C4.5 algorithm.

Table 10
kNN results for plain data, DAE and ClEnDAE models (AUC).

Dataset	Plain Data	DAE	ClEnDAE
arcene	0.693	0.672	**0.719**
batch	0.870	0.990	**0.992**
coil2000	0.546	0.554	**0.562**
dota	0.416	0.522	**0.534**
drive	0.800	0.846	**0.879**
facial	**0.795**	0.685	0.698
fashionmnist	0.906	0.920	**0.929**
gisette	0.824	**0.923**	**0.923**
hapt	0.903	0.931	**0.942**
image	0.929	0.950	**0.954**
isolet	0.943	0.968	**0.975**
letter	**0.973**	0.923	0.951
madelon	0.526	0.591	**0.614**
mfeat	0.703	0.984	**0.988**
microv1	0.920	0.939	**0.939**
microv2	0.883	0.932	**0.951**
mnist	0.965	0.975	**0.978**
musk	0.829	0.941	**0.947**
nomao	0.891	0.904	**0.914**
semeion	0.927	0.940	**0.945**

Table 11
SVM results for plain data, DAE and ClEnDAE models (AUC).

Dataset	Plain Data	DAE	ClEnDAE
arcene	0.500	0.632	**0.642**
batch	0.923	0.980	**0.985**
coil2000	**0.500**	**0.500**	**0.500**
dota	0.509	0.550	**0.553**
drive	0.885	0.941	**0.972**
facial	**0.802**	0.713	0.742
fashionmnist	0.932	0.940	**0.943**
gisette	0.803	0.969	**0.972**
hapt	0.900	0.889	**0.902**
image	0.846	0.877	**0.891**
isolet	0.954	0.975	**0.980**
letter	**0.966**	0.828	0.861
madelon	0.569	0.591	**0.594**
mfeat	0.970	**0.985**	**0.985**
microv1	0.500	0.918	**0.925**
microv2	0.500	0.871	**0.893**
mnist	0.981	0.985	**0.986**
musk	0.838	0.948	**0.962**
nomao	0.914	0.924	**0.935**
semeion	0.891	0.957	**0.959**

Table 12
MLP results for plain data, DAE and ClEnDAE models (AUC).

Dataset	Plain Data	DAE	ClEnDAE
arcene	0.522	0.500	**0.525**
batch	0.914	0.920	**0.926**
coil2000	**0.500**	**0.500**	**0.500**
dota	0.500	0.500	**0.538**
drive	0.698	0.680	**0.772**
facial	0.520	0.701	**0.716**
fashionmnist	0.901	0.911	**0.914**
gisette	0.605	0.627	**0.682**
hapt	0.839	0.850	**0.857**
image	0.703	0.872	**0.888**
isolet	0.827	0.842	**0.869**
letter	0.734	0.740	**0.748**
madelon	0.500	0.637	**0.641**
mfeat	0.500	0.968	**0.975**
microv1	0.678	0.690	**0.711**
microv2	0.708	0.725	**0.742**
mnist	0.872	0.905	**0.906**
musk	0.850	**0.999**	**0.999**
nomao	**0.901**	0.530	0.653
semeion	**0.827**	0.756	0.789

Table 13
C4.5 results for plain data, DAE and ClEnDAE models (AUC).

Dataset	Plain Data	DAE	ClEnDAE
arcene	0.655	0.678	**0.699**
batch	0.921	0.982	**0.984**
coil2000	0.503	0.516	**0.521**
dota	0.462	0.539	**0.546**
drive	0.869	0.850	**0.884**
facial	**0.789**	0.655	0.672
fashionmnist	0.877	0.888	**0.889**
gisette	0.657	0.743	**0.749**
hapt	0.798	0.799	**0.802**
image	0.832	0.859	**0.874**
isolet	0.870	0.874	**0.878**
letter	**0.922**	0.811	0.821
madelon	0.520	0.524	**0.553**
mfeat	0.911	0.913	**0.931**

Table 13 (*continued*)

Dataset	Plain Data	DAE	ClEnDAE
microv1	0.838	0.849	**0.880**
microv2	0.736	0.845	**0.863**
mnist	0.965	0.975	**0.977**
musk	0.800	0.883	**0.886**
nomao	0.886	0.877	**0.889**
semeion	0.716	0.739	**0.751**

Table 14
kNN classification results of ClEnDAE, PCA, LDA, ISOMAP and LLE for test data (AUC).

Dataset	PCA	LDA	ISOMAP	LLE	ClEnDAE
arcene	0.661	0.654	0.738	**0.747**	0.719
batch	0.861	0.852	0.786	0.862	**0.992**
coil2000	0.523	0.525	0.525	0.502	**0.562**
dota	0.513	0.517	0.516	0.516	**0.534**
drive	0.683	0.782	0.693	0.726	**0.879**
facial	0.631	0.648	0.601	0.602	**0.698**
fashionmnist	0.831	0.911	0.842	0.896	**0.929**
gisette	0.855	0.883	0.877	0.878	**0.923**
hapt	0.553	0.903	0.750	0.781	**0.942**
image	0.734	0.779	0.882	0.874	**0.954**
isolet	0.659	0.971	0.962	0.927	**0.975**
letter	0.882	0.893	0.894	0.883	**0.951**
madelon	0.523	0.505	0.506	0.501	**0.614**
mfeat	0.963	0.972	0.985	0.986	**0.988**
microv1	0.532	0.897	**0.947**	0.946	0.939
microv2	0.632	0.891	0.948	**0.951**	**0.951**
mnist	0.828	0.966	0.923	0.944	**0.978**
musk	0.912	**0.964**	0.938	0.901	0.947
nomao	0.797	0.817	0.804	0.849	**0.914**
semeion	0.732	0.926	0.894	0.884	**0.945**

Table 15
SVM classification results of ClEnDAE, PCA, LDA, ISOMAP and LLE for test data (AUC).

Dataset	PCA	LDA	ISOMAP	LLE	ClEnDAE
arcene	0.531	0.545	0.721	**0.722**	0.642
batch	0.983	0.980	0.983	0.892	**0.985**
coil2000	0.493	**0.500**	**0.500**	**0.500**	**0.500**
dota	0.517	0.529	0.547	0.548	**0.553**
drive	0.978	0.979	0.981	**0.982**	0.972
facial	0.672	0.703	0.697	0.501	**0.742**
fashionmnist	0.921	0.935	0.929	0.934	**0.943**
gisette	0.925	0.941	0.962	0.963	**0.972**
hapt	0.788	0.875	0.859	0.869	**0.902**
image	0.850	0.862	0.784	0.620	**0.891**
isolet	0.958	0.965	0.963	0.966	**0.980**
letter	0.812	0.839	0.842	0.852	**0.861**
madelon	0.557	0.585	0.547	0.523	**0.594**
mfeat	0.926	0.962	0.980	0.981	**0.985**
microv1	0.731	0.876	**0.953**	**0.953**	0.925
microv2	0.652	0.791	0.832	0.835	**0.893**
mnist	0.971	0.979	0.829	0.976	**0.986**
musk	0.955	0.958	0.959	0.954	**0.962**
nomao	0.929	0.930	0.933	0.925	**0.935**
semeion	0.936	0.957	0.929	0.949	**0.959**

Table 16
MLP classification results of ClEnDAE, PCA, LDA, ISOMAP and LLE for test data (AUC).

Dataset	PCA	LDA	ISOMAP	LLE	ClEnDAE
arcene	0.521	0.512	**0.691**	0.673	0.525
batch	0.915	0.512	0.919	0.566	**0.926**
coil2000	**0.500**	**0.500**	**0.500**	**0.500**	**0.500**
dota	0.524	0.525	0.527	0.525	**0.538**
drive	**0.941**	0.883	0.802	0.809	0.772
facial	0.692	0.704	0.707	0.697	**0.716**
fashionmnist	0.902	0.909	0.898	0.905	**0.914**
gisette	0.602	0.612	0.609	0.605	**0.682**
hapt	0.834	0.839	0.840	0.836	**0.857**
image	0.881	0.552	0.672	0.645	**0.888**
isolet	0.849	0.819	0.791	0.805	**0.869**
letter	0.713	0.732	0.739	0.722	**0.748**
madelon	0.577	0.582	0.556	0.512	**0.641**
mfeat	0.921	0.953	0.974	0.964	**0.975**
microv1	0.694	0.701	0.829	**0.868**	0.711
microv2	0.731	0.720	0.740	0.733	**0.742**
mnist	0.902	**0.906**	0.893	0.896	**0.906**
musk	0.992	0.991	0.994	0.952	**0.999**
nomao	0.913	**0.915**	0.911	0.914	0.653
semeion	0.820	**0.849**	0.746	0.822	0.789

Table 17
C4.5 classification results of ClEnDAE, PCA, LDA, ISOMAP and LLE for test data (AUC).

Dataset	PCA	LDA	ISOMAP	LLE	ClEnDAE
arcene	0.654	0.662	0.766	0.695	**0.699**
batch	0.971	0.973	0.980	0.975	**0.984**
coil2000	0.509	0.504	0.507	0.500	**0.521**
dota	0.527	0.531	0.503	0.515	**0.546**
drive	0.936	0.938	**0.954**	0.947	0.884
facial	0.626	0.651	0.635	0.632	**0.672**
fashionmnist	0.872	0.883	0.879	0.881	**0.889**
gisette	0.682	0.705	0.692	0.709	**0.749**
hapt	0.797	0.791	0.792	0.788	**0.802**
image	0.913	**0.922**	0.774	0.838	0.874
isolet	0.840	0.859	0.851	0.862	**0.878**
letter	0.783	0.811	0.793	0.805	**0.821**
madelon	0.626	**0.677**	0.652	0.631	0.553
mfeat	0.889	0.898	0.892	0.895	**0.931**
microv1	0.731	0.806	0.685	0.868	**0.880**
microv2	0.835	0.837	0.815	0.838	**0.863**
mnist	0.869	0.872	0.876	0.871	**0.977**
musk	0.891	**0.927**	0.897	0.915	0.886
nomao	0.881	0.862	0.767	0.872	**0.889**
semeion	0.729	0.739	0.744	0.731	**0.751**

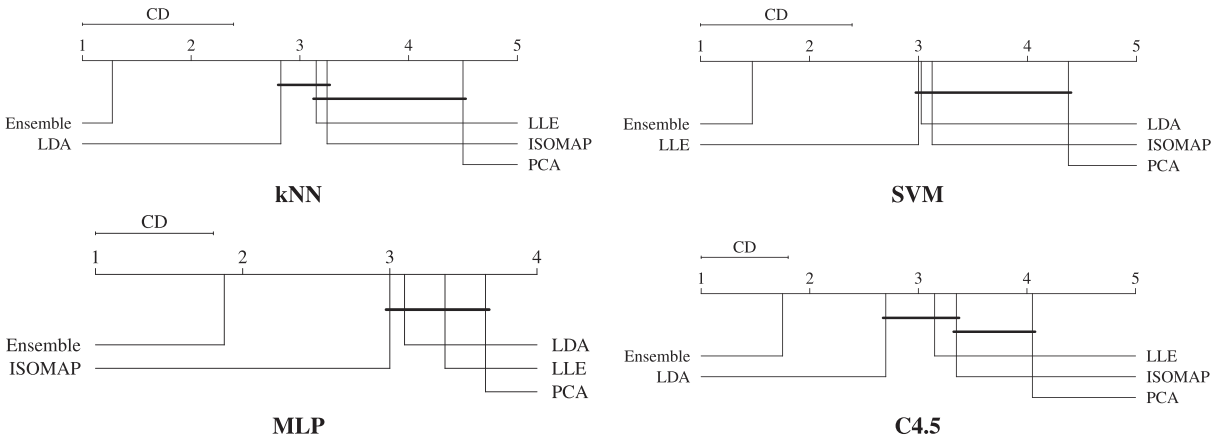

Fig. 17. Critical distance between dimensionality reduction methods for kNN, SVM, MLP and C4.5.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

The work of F. Pulgar was supported by the Spanish Ministry of Education under the FPU National Program (Ref. FPU16/00324). This work was partially supported by the Spanish Ministry of Science and Technology under project TIN2015-68454-R and by the Spanish Ministry of Science and Innovation under project PID2019-107793GB-I00 / AEI / 10.13039/501100011033.

Appendix A. Result tables

This appendix contains the results tables associated with the experimentation presented in Section 4. In particular, A.1 includes the results tables of SubSection 4.2, and A.2 presents the data associated with the experimentation described in SubSection 4.4.

A.1. Classification Algorithms Analysis

This appendix provides Tables 10–13. The data describe the results achieved in the experimentation for the algorithms kNN, SVM, MLP and C4.5, respectively. The structure of all tables is similar. The classification results (AUC) generated with the ClEnDAE model, the basic DAE and the raw data are presented in a figure, and the best results are highlighted in bold for each dataset.

Tables 10–13 allow us to observe that the best performance is obtained with the ClEnDAE method in most cases. These results are discussed in SubSection 4.2 in more detail.

A.2. ClEnDAE vs classical feature extraction techniques

In this appendix, classification results for the different datasets are gathered in Tables 14–17. There is a table for each classifier: kNN, SVM, MLP and C4.5, respectively. Besides, each one presents the predictive performance generated from the following models: the proposal of this study, ClEnDAE, and the traditional dimensionality reduction approaches, PCA, LDA, ISOMAP and LLE.

These outcomes have been analyzed in Section 4.2. Generally, it is possible to note that the ClEnDAE algorithm obtains the best results in almost every case, considering the four classifiers.

References

[1] S.B. Kotsiantis, Supervised machine learning: A review of classification techniques, Informatica 31 (2007) 249–268.
[2] L.J.P. Van Der Maaten, E.O. Postma, H.J. Van Den Herik, Dimensionality Reduction: A Comparative Review, Journal of Machine Learning Research 10 (2009) 1–41, doi:10.1.1.112.5472.
[3] F.J. Pulgar, F. Charte, A.J. Rivera, M.J. del Jesus, Choosing the proper autoencoder for feature fusion based on data complexity and classifiers: Analysis, tips and guidelines, Information Fusion 54 (2020) 44–60, https://doi.org/10.1016/j.inffus.2019.07.004.
[4] S. Bengio, Y. Bengio, Taking on the curse of dimensionality in joint distributions using neural networks, IEEE Transactions on Neural Networks 11 (3) (2000) 550–557, https://doi.org/10.1109/72.846725.
[5] R. Bellman, Adaptive control processes: A guided tour, Princeton University Press, 1961.
[6] M. Galar, A. Fernandez, E. Barrenechea, H. Bustince, F. Herrera, A review on ensembles for the class imbalance problem: bagging-, boosting-, and hybrid-based approaches, IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews) 42 (4) (2012) 463–484.
[7] K. Pearson, LIII. On lines and planes of closest fit to systems of points in space, The London, Edinburgh, and Dublin Philosophical Magazine and Journal of, Science 2 (11) (1901) 559–572, https://doi.org/10.1080/14786440109462720.
[8] H. Yu, J. Yang, A direct LDA algorithm for high-dimensional data-with application to face recognition, Pattern Recognition 34 (10) (2001) 2067–2070, https://doi.org/10.1016/S0031-3203(00)00162-X.
[9] J.B. Tenenbaum, A Global Geometric Framework for Nonlinear Dimensionality Reduction, Science 290 (5500) (2000) 2319–2323, https://doi.org/10.1126/science.290.5500.2319.
[10] S.T. Roweis, L.K. Saul, Nonlinear dimensionality reduction by locally linear embedding, Science 290 (5500) (2000) 2323–2326, https://doi.org/10.1126/science.290.5500.2323.
[11] S. García, J. Luengo, F. Herrera, Feature Selection, Springer International Publishing, Cham (2015) 163–193, https://doi.org/10.1007/978-3-319-10247-4_7.
[12] D. Charte, F. Charte, S. García, M.J. del Jesus, F. Herrera, A practical tutorial on autoencoders for nonlinear feature fusion: Taxonomy, models, software and guidelines, Information Fusion 44 (2018) 78–96, https://doi.org/10.1016/j.inffus.2017.12.007.
[13] U.G. Mangai, S. Samanta, S. Das, P.R. Chowdhury, A survey of decision fusion and feature fusion strategies for pattern classification, IETE Technical review 27 (4) (2010) 293–307, https://doi.org/10.4103/0256-4602.64604.
[14] I. Goodfellow, Y. Bengio, A. Courville, Deep Learning, The MIT Press, 2016.
[15] A. Krizhevsky, I. Sutskever, H. Geoffrey E., ImageNet Classification with Deep Convolutional Neural Networks, in: Advances in neural information processing systems, 2012, pp. 1097–1105..
[16] G.E. Hinton, R.R. Salakhutdinov, Reducing the Dimensionality of Data with Neural Networks, Science 313 (5786) (2006) 504–507, https://doi.org/10.1126/science.1127647.

[17] Y. Bengio, Deep Learning of Representations: Looking Forward, International Conference on Statistical Language and Speech Processing (2013) 1–37, https://doi.org/10.1007/978-3-642-39593-2_1.
[18] F.J. Pulgar, F. Charte, A.J. Rivera, M.J.D. Jesus, AEkNN An AutoEncoder kNN-Based Classifier With Built-in Dimensionality Reduction, International Journal of Computational Intelligence Systems 12 (2018) 436–452, https://doi.org/10.2991/ijcis.2018.125905686.
[19] F.J. Pulgar, F. Charte, A.J. Rivera, M.J. del Jesus, A first approach to face dimensionality reduction through denoising autoencoders, in: International Conference on Intelligent Data Engineering and Automated Learning, Springer, 2018, pp. 439–447.
[20] D. García-Gil, S. Ramírez-Gallego, S. García, F. Herrera, On the Use of Random Discretization and Dimensionality Reduction in Ensembles for Big Data (2018) 15–26, https://doi.org/10.1007/978-3-319-92639-1_2.
[21] J.J. Rodriguez, L.I. Kuncheva, C.J. Alonso, Rotation forest: A new classifier ensemble method, IEEE transactions on pattern analysis and machine intelligence 28 (10) (2006) 1619–1630, https://doi.org/10.1109/TPAMI.2006.211.
[22] R.F. Alvear-Sandoval, J.L. Sancho-Gómez, A.R. Figueiras-Vidal, On improving cnns performance: The case of mnist, Information Fusion 52 (2019) 106–109.
[23] T. Cover, P. Hart, Nearest neighbor pattern classification, IEEE Transactions on Information Theory 13 (1) (1967) 21–27, https://doi.org/10.1109/TIT.1967.1053964.
[24] B.E. Boser, I.M. Guyon, V.N. Vapnik, A training algorithm for optimal margin classifiers, in: Proceedings of the fifth annual workshop on Computational learning theory, ACM, 1992, pp. 144–152..
[25] K. Hornik, M. Stinchcombe, H. White, Multilayer feedforward networks are universal approximators, Neural networks 2 (5) (1989) 359–366.
[26] J.R. Quinlan, Induction of decision trees, Machine learning 1 (1) (1986) 81–106.
[27] C.P. Chen, C.-Y. Zhang, Data-intensive applications, challenges, techniques and technologies: A survey on big data, Information Sciences 275 (2014) 314–347, https://doi.org/10.1016/j.ins.2014.01.015.
[28] M. Dash, H. Liu, Feature selection for classification, Intelligent data analysis 1 (3) (1997) 131–156, https://doi.org/10.3233/IDA-1997-1302.
[29] I. Guyon, S. Gunn, A. Ben-Hur, G. Dror, Result analysis of the nips, feature selection challenge, Advances in neural information processing systems 2005 (2003) 545–552.
[30] D.W. Aha, D. Kibler, M.K. Albert, Instance-Based Learning Algorithms, Machine Learning 6 (1) (1991) 37–66, https://doi.org/10.1023/A:1022689900470.
[31] C.G. Atkeson, A.W. Moorey, S. Schaalz, A.W. Moore, S. Schaal, Locally Weighted Learning, Artificial Intelligence 11 (1997) 11–73, https://doi.org/10.1023/A:1006559212014.
[32] M.A. Hearst, S.T. Dumais, E. Osuna, J. Platt, B. Scholkopf, Support vector machines, IEEE Intelligent Systems and their applications 13 (4) (1998) 18–28.
[33] R.J. Schalkoff, Artificial neural networks, Vol. 1, McGraw-Hill New York, 1997..
[34] T. Wang, J. Qiu, H. Gao, Adaptive neural control of stochastic nonlinear time-delay systems with multiple constraints, IEEE Transactions on Systems, Man, and Cybernetics: Systems 47 (8) (2017) 1875–1883, https://doi.org/10.1109/TSMC.2016.2562511.
[35] X. Yu, T. Wang, J. Qiu, H. Gao, Barrier lyapunov function-based adaptive fault-tolerant control for a class of strict-feedback stochastic nonlinear systems, IEEE Transactions on Cybernetics (2019) 1–9, https://doi.org/10.1109/TCYB.2019.2941367.
[36] M.W. Gardner, S. Dorling, Artificial neural networks (the multilayer perceptron)-a review of applications in the atmospheric sciences, Atmospheric environment 32 (14–15) (1998) 2627–2636.
[37] Y. Qi, Y. Wang, X. Zheng, Z. Wu, Robust feature learning by stacked autoencoder with maximum correntropy criterion, in: in: 2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2014, pp. 6716–6720, https://doi.org/10.1109/ICASSP.2014.6854900.
[38] S. Rifai, X. Muller, Contractive Auto-Encoders: Explicit Invariance During Feature Extraction, in: Proceedings of the 28th International Conference on Machine Learning, Vol. 85, 2011, pp. 833–840..
[39] P. Vincent, H. Larochelle, Y. Bengio, P.-A. Manzagol, Extracting and composing robust features with denoising autoencoders, in, in: Proceedings of the 25th International Conference on Machine Learning ACM, 2008, pp. 1096–1103, https://doi.org/10.1145/1390156.1390294.
[40] Y. Meng, R. Shang, F. Shang, L. Jiao, S. Yang, R. Stolkin, Semi-supervised graph regularized deep nmf with bi-orthogonal constraints for data representation, IEEE Transactions on Neural Networks and Learning Systems (2019) 1–14, https://doi.org/10.1109/TNNLS.2019.2939637.
[41] R. Shang, W. Wang, R. Stolkin, L. Jiao, Non-negative spectral learning and sparse regression-based dual-graph regularized feature selection, IEEE Transactions on Cybernetics 48 (2) (2018) 793–806, https://doi.org/10.1109/TCYB.2017.2657007.
[42] R. Shang, Y. Meng, W. Wang, F. Shang, L. Jiao, Local discriminative based sparse subspace learning for feature selection, Pattern Recognition 92 (2019) 219–230, https://doi.org/10.1016/j.patcog.2019.03.026.
[43] Y. Meng, R. Shang, L. Jiao, W. Zhang, Y. Yuan, S. Yang, Feature selection based dual-graph sparse non-negative matrix factorization for local discriminative clustering, Neurocomputing 290 (2018) 87–99, https://doi.org/10.1016/j.neucom.2018.02.044.
[44] D.J. Hand, R.J. Till, A simple generalisation of the area under the roc curve for multiple class classification problems, Machine Learning 45 (2) (2001) 171–186, https://doi.org/10.1023/A:1010920819831.
[45] M. Friedman, The use of ranks to avoid the assumption of normality implicit in the analysis of variance, Journal of the American Statistical Association 32 (200) (1937) 675–701, https://doi.org/10.1080/01621459.1937.10503522.
[46] J.D. Li, A two-step rejection procedure for testing multiple hypotheses, Journal of Statistical Planning and Inference 138 (6) (2008) 1521–1527, https://doi.org/10.1016/j.jspi.2007.04.032.
[47] D. Meyer, E. Dimitriadou, K. Hornik, A. Weingessel, F. Leisch, e1071: Misc Functions of the Department of Statistics, Probability Theory Group (Formerly: E1071), TU Wien, r package version 1.6-7 (2015). URL: https://CRAN.R-project.org/package=e1071..
[48] M.K.C. from Jed Wing, S. Weston, A. Williams, C. Keefer, A. Engelhardt, T. Cooper, Z. Mayer, B. Kenkel, the R Core Team, M. Benesty, R. Lescarbeau, A. Ziem, L. Scrucca, Y. Tang, C. Candan., caret: Classification and Regression Training, r package version 6.0-68 (2016). URL: https://CRAN.R-project.org/package=caret..
[49] K. Hornik, C. Buchta, A. Zeileis, Open-source machine learning: R meets weka, Computational Statistics 24 (2) (2009) 225–232, https://doi.org/10.1007/s00180-008-0119-7.
[50] N. Miloslavskaya, A. Tolstoy, Big data, fast data and data lake concepts, Procedia Computer Science 88 (300–305) (2016) 63.