Alvear Sandoval, R. F. y Figueiras Vidal, A. R. (2018). On building ensembles of stacked denoising auto-encoding classifiers and their further improvement. *Information Fusion*, 39, pp. 41-52.

DOI: https://doi.org/10.1016/j.inffus.2017.03.008

On building ensembles of stacked denoising auto-encoding classifiers and their further improvement

Ricardo F. Alvear-Sandoval *, Aníbal R. Figueiras-Vidal
GAMMA-L+/DTSC, Universidad Carlos III de Madrid, Spain

abstract

To aggregate diverse learners and to train deep architectures are the two principal avenues towards in- creasing the expressive capabilities of neural networks. Therefore, their combinations merit attention. In this contribution, we study how to apply some conventional diversity methods –bagging and label switching– to a general deep machine, the stacked denoising auto-encoding classifier, in order to solve a number of appropriately selected image recognition problems. The main conclusion of our work is that binarizing multi-class problems is the key to obtain benefit from those diversity methods.

Additionally, we check that adding other kinds of performance improvement procedures, such as pre- emphasizing training samples and elastic distortion mechanisms, further increases the quality of the re- sults. In particular, an appropriate combination of all the above methods leads us to reach a new absolute record in classifying MNIST handwritten digits.

These facts reveal that there are clear opportunities for designing more powerful classifiers by means of combining different improvement techniques.

Keywords: Augmentation; Classification; Deep; Diversity; Learning; Pre-emphasis

1. Introduction

The number of available training samples limits the expressive capability of traditional one-hidden layer perceptrons, or shallow Multi-Layer Perceptrons (MLPs), for practical applications, in spite of their theoretically unbounded approximation capacities [1,2]. Consequently, a lot of attention is being paid to architecture and parameterization procedures that allow to improve their performance when solving practical problems. The most relevant procedures increase the number of the trainable weights following two main avenues: Building ensembles of learning machines, or constructing Deep Neural Networks (DNNs).

Most of the advances in DNN design correspond to the last decade. In fact, prior to 2006 the only successfully used DNNs were the Convolutional Neural Network (CNN) classifiers [3], whose significantly simplified structure makes possible their training by means of conventional algorithms such as Back Propagation (BP). This architecture is appropriate for some kinds of applications, those in which the samples show translation-invariant characteristics, for example, image processing. But direct training of general form DNNs remained without solution because the appearance of vanishing or exploding derivatives [4,5]. In 2006 Hinton et al. [6] proposed a deep classifier indirect design that involved the stacking of Reduced Boltzmann Machines (RBMs) [7]. A top layer task-oriented training and overall refining complete these classifiers, that have come to be called Deep Belief Machines (DBMs). Contrastive divergence algorithms allow for the training of the DBMs without a huge computational effort [8].

Some years later, Vincent et al. [9] introduced a similar procedure consisting of an expansive, denoising auto-encoder layer-wise training plus the final top classification and refining. These are the Stacked Denoising Auto-Encoder (SDAE) classifiers. It is worth mentioning that both DBMs and SDAEs are representation machines [10], i.e., their hidden layers provide more and more sophisticated high-level feature representations of the input vectors. These representations can be useful for analysis purposes [11], and, even more, the representation process induces a disentangling of the sub-spaces in which the samples appear [12]. On the other hand, another sequentially trainable deep architecture, the Deep Stacking Networks (DSNs), was introduced in [13,14], following the idea of training shallow MLPs and adding their outputs to the input vector for training further units. Finally, a number of modifications that reduce the difficulties with the derivatives have been proposed for training directly DNNs, such as data conscious initializations [15], Hessian-free search [16], mini-batch iterations [17], non-sigmoidal activations [18], and adding scale and location trainable parameters [19].

There are also proofs of universal approximation capabilities for DNNs [20,21], as well as of some interesting characteristics of them

* Corresponding author.
E-mail address: ralvear@tsc.uc3m.es (R.F. Alvear-Sandoval).

[22]. The analyses in [23,24] show that by adding layers to a network, it is possible to reduce the effort to establish input-output correspondences. In practice, DNNs have offered excellent performance results in many applications, therefore, one can conclude they are important in spite of the large number of parameters that need to be learned. There is not room here to give more details, so, the interested reader is referred to excellent tutorials [4,5,25] for more extensive reviews and bibliography, as well as to [26] for a bibliography of applications.

Ensembles are the second option to effectively increase the expressive capability of learning machines, including MLPs. They are built by means of training learners that consider the problem to be solved from different perspectives, i.e., under a principle of diversity, and aggregating their outputs to obtain an improved solution. We present a very concise overview of ensembles, emphasizing just the design methods that we will use in our experiments, in Section 2, for the sake of continuity in this Introduction.

Since both diversity and depth increase the expressive power of MLPs but through very different mechanisms, it seems reasonable to expect that a combination of them would lead to an even better performance. However, there are a moderate number of contributions along this research direction. We will briefly revise them in Section 4, but we anticipate that some difficulties appear when trying to apply the usual ensemble building methods to multi-class problems, and, consequently, most of the DNN ensembles are constructed by means of "ad hoc" procedures.

In this paper, we explore and discuss in detail how and why diversification can be applied to DNNs, as well as if including other improvement techniques gives additional advantage. The objective is to evaluate if it is possible to get significant advantages by combining diversification and deep learning, as well as other techniques.

Of course, we have to select both DNN architectures and classification problems for our experiments and subsequent analysis. Although most of the previous studies with the databases we will use have considered CNNs, we have decided to work with a less specific architecture to exclude the possibility of obtaining conclusions only valid for this particular form of DNN and the kind of problems that are appropriate for it. So, we select SDAE classifiers, and in particular the SDAE-3 design that is introduced in [9]. However, at the same time and just to show the potential of combining diversity and depth, we will address some traditional image classification tasks –also included in [9],– that are more appropriate for CNN architectures. The selected problems for our experiments will be the well-known 10-class handwritten digit MNIST database [3], its version with a smaller training set MNIST-BASIC [9], in order to analyze the relevance of the weak or strong character of the SDAE-3 classifiers, and also the binary database RECTANGLES [9], with the objective of studying the origin of the difficulties for creating ensembles of multi-class DNNs. We emphasize that these selections are not arbitrary: There are many published results for MNIST, for example in [27,28], and clearly established records for representation DNNs, a 0.86% error rate [10], and for CNN ensembles [29], a 0.21% error rate. We anticipate from now that, with the help of a boosting-type training reinforcement or pre-emphasis, and a simple data augmentation besides of the binarization and training diversification we will apply, we arrive to a new absolute performance record, a 0.19% error rate. We repeat that this record was not our objective, but we looked for a better understanding of how to combine diversity and depth, and to avoid conclusions only valid for particular situations –using CNNs for image problems,– we select both SDAEs and the databases. Thus, in our opinion, there is no reason to think that our conclusions are problem- or architecture-dependent.

The rest of the paper is structured as follows. In Section 2, we present brief overviews of machine ensembles, both general forms and those that come from binarizing multi-class problems. We dedicate Section 4 to list and comment previously published works in designing DNN ensembles. Section 5 describes the additional techniques –pre-emphasis and data augmentation– we will use in the second part of our experiments. The experimental framework is detailed in Section 6: Databases, deep learning units, diversification and binarization techniques, and pre-emphasis and data augmentation forms. The results of the experiments appear in Section 7 following a sequential order, plus the corresponding discussions. Finally, the main conclusions of this work and some directions for further research close the paper.

2. Ensembles

To build an ensemble of diverse machines and aggregate their outputs is a way to increase expressive power. Although the first ideas on it were published half a century ago [30], they have been mainly developed along the last two decades. In the following, we briefly review some ensemble techniques, including those we will use to diversify SDAEs. We dedicate separate sub-sections to designs that introduce diversification by means of architecture or training differences, that we will call conventional ensembles, and to ensembles that come from transforming a multi-class problem in a number of binary classifications from which the resulting class can be obtained. Since a complete review of ensembles is beyond the scope of this paper, the reader is referred to monographs [31–34], as well as to tutorial article [35], which includes interesting perspectives on ensemble applications.

2.1. Conventional ensembles

Conventional diversification methods may be broadly classified into two categories. The first are those approaches that independently train a number of machines, usually with different training sets. These machines, or learners, can also have different structures. After it, learners' outputs are aggregated –typically with simple, non-trainable procedures– to come up with the final classification. These ensembles are called committees.

Among committees, Random Forests (RFs) [36] are very popular because they offer a remarkable performance. They diversify a number of tree classifiers by means of probabilistic branching, which can be combined with sub-space projections. There are other committees that can be applied to general types of learners, requiring only that they are unstable: Bagging [37] and label switching [38,39]. We will include both of them in our experiments because they are simple to implement and provide high expressive power, clearly improving the performance of a single machine. Yet we announce that the first experimental results will lead us to focus on the second.

Bagging ("Bootstrap and aggregating") produces diversity by training the ensemble learners with bootstraping re-sampled versions of the original training set and, then, aggregating these learners' outputs, usually by averaging them or with a majority vote. Bootstrap is a random sampling mechanism which includes replacement to permit arbitrary sizes of the re-sampled population. Although its primitive form used bootstrapped sets of the same size as the true training set, to explore the size of these bootstrapped sets is important to find a good balance between computational effort and number of learners and ensemble performance, because in some cases the reduction of the true samples that each learner sees can provoke losses. On the other hand, label switching changes the labels of a given portion of the training samples according to some stochastic mechanism. We will employ the simplest version, for which these changes appear purely at random. The switching rate must be explored when designing these committees.

The second type of conventional ensembles, which we call consortia, are algorithms that train both the learners and the aggregation in a related manner. The best known method in this category is boosting, which has proven to provide excellent classification performances by combining the outputs of weak learners [40,41]. Its principle is to sequentially design and aggregate weak learners that, at each stage, pay more attention to the examples that have larger classification error according to the previously built partial ensemble. Many extensions and generalizations of the basic boosting algorithms have been proposed [34]. Among them, we mention [42,43] because they also consider the proximity to the classification boundary of the training examples, and this idea is in the origin of the formulas for the pre-emphasis techniques we will apply in this paper, that will be introduced in Section 5. But, since boosting methods require weak learners, we do not include them in our experiments to diversify SDAE classifiers.

Other consortia algorithms are those known as Negative Correlation Learning (NCL) ensembles [44,45] and the Mixtures of Experts (MoEs) [46]. Both of them could be applied to diversify DNNs, but their performance improvements are moderate for classification, and their many modifications to get more advantage [47,48] demand higher computational effort, which, in principle, is not ideal when looking for methods to diversify DNNs.

According to the above, we will adopt in our experiments bagging and label switching as conventional diversification mechanisms, although we will also apply pre-emphasis forms that are conceptually equivalent to generalized boosting weighting.

2.2. Ensembles of binary classifiers for multi-class problems

A different type of ensemble is that formed by decomposing a multi-class problem into a number of binary problems [33]. Although of different origin and constructed in a different manner that conventional ensembles, binarization techniques are true ensembles since they consist of a collection of binary machines, each one having a different function or task that is related to the overall problem of multi-class classification, and the outputs of the binary machines are aggregated to produce a classification that is better than any one machine used alone.

There are two basic binarization techniques, One versus One (OvO) and One versus Rest (OvR) [49,50]. For a C-class problem, OvO is an ensemble of $C(C-1)/2$ binary classifiers that perform pair-wise classifications of one class C_j versus another class C_k for each $j \neq k$. The class that has the most votes is then determined to be the correct class. OvR is an ensemble of C binary classifiers where each classifier makes a decision between one class and the rest. The advantages of OvR are that the number of learners is smaller (obviously, C) –although it may be argued that this leads to less diversity– and that each classifier sees all of the training samples. However, its disadvantage is that the learner data sets become imbalanced and this fact tends to produce a decrease in the performance of individual classifiers and, consequently, of the whole ensemble. Although this can be alleviated by using carefully designed re-balancing mechanisms, we decide to use OvO to avoid the corresponding risks.

Another approach to the binarization of a multi-class problem that is more effective than OvO and OvR are the Error Correcting Output Codes (ECOCs) [51]. ECOCs are typically better than OvO or OvR because a wrong decision requires a number of wrong binary classifications –see the details below,– and not just a wrong majority (OvO) or a wrong high output level (OvR). However, effective ECOC designs are very difficult for many class problems. More details can be found in [33]. An ECOC binarization associates a binary codeword to each class, and the resulting binary problems are those represented by the columns, each dichotomy being formed by grouping the true classes according to their correspondence to 0 or 1 bits. The overall classification is performed by finding the class whose codeword is the closest to the one that is produces by the ensemble of classifiers. This implies that codewords with high Hamming distances must be selected, and, consequently, there is a clear compromise with the length of the codeword, i.e., the size of the ECOC ensemble. In our experiments, we will use the 15-bit, 10-class ECOC which was introduced in [51] just for MNIST considering different characteristics of handwritten digits.

We must say that, when the number of classes is high, OvO schemes become impractical, and ECOC ensembles are difficult to design. Monograph [33] provides some details about how to do it. To consistently re-balance OvR dichotomies is a good alternative, and the same is useful for big ECOC ensembles. Since we do not deal here with such a kind of problems, we do not further discuss this subject.

3. A concise overview of previous approaches to diversify DNNs

The first CNN ensemble was presented in [52], with different image sizing as the diversification technique. Simple data augmentation mechanisms allowed a record performance for MNIST, an excellent 0.21% error rate, using again a CNN ensemble [29]. Direct averaging of classification outputs obtained from consecutive hidden layers and/or of the top output at different training epochs is proposed in [53], with moderate advantages. In GoogLeNet [54], bagging is applied together with different weight initializations to get a very deep and powerful CNN machine. In [55], trainable simple aggregation schemes are applied to CNN ensembles to further improve their performance. Optical flow obtained from consecutive video frames is used in [56] to construct CNN ensembles. Similarly, multiple triphone states are the source of diversity for speech recognition purposes in [57], using learners that include hidden Markov model units. The authors of [58] use spectral diversity [59] to train some DNNs whose outputs are aggregated, while partial training of learners with distorted data sets in the diversification applied in [60].

From our point of view, the studies carried out in [61] are specially relevant. Their authors found that there are difficulties to apply bagging to CNNs: In fact, average performances for several databases become worse than applying only random initializations. They also found that to diversify layers that are near to the final output is more effective, a fact that we also observed at the some time for SDAE-3 learners [62]. These were the starting points for the research we present in this paper.

We conclude this concise review mentioning two very recent contributions: Simultaneously training aggregation weights and learners for several DNN architectures, including CNNs [63], and selectively combining CNNs that have been trained with powerful augmentation procedures [28]. Finally, let us clarify that we do not discuss here important methods such as drop-out [64] and drop-connect [65] because, although some researchers consider them as forms of diversification, we firmly support that they are probabilistic regularization methods (that subsequently produce some kind of elementary diversification), since their main effect is to reduce the possibility of overfitting by randomly limiting the active units or weights at each learning algorithm step.

An important conclusion can be extracted from the above review: To apply conventional diversification techniques to DNNs is not an easy task, and this seems to be the reason why most of the DNN ensembles are built using other diversification mechanisms, such as “ad hoc” image augmentation pre-processing, speech sequential features, or even partial training or manual selection. Since our experience indicates that the case is similar to that of shallow MLPs applied to multi-class problems, where binarization not only provides a direct advantage but it also makes conventional diversification more effective, we oriented the first phase of our re-

search to check if this is also true for DNNs, in particular, for SDAE classifiers, that we selected by the reasons we indicated in the Introduction. After obtaining good results, we decided to continue by adding other complementary procedure that also much improves the classification performance of several kinds of shallow machines (including shallow MLPs), pre-emphasis, under a general formulation we will introduce in the next section. And, finally, given the success that the application of data augmentation pre-processing has demonstrated in many experiments, we added a simple form of it –also described in Section 5– to the combination of binarization, conventional diversification, and pre-emphasis. The results have been excellent, including, as announced above, a new absolute record for the MNIST database, a 0.19% average error rate, in spite of using a non-convolutional basic learner, the SDAE-3 classifier. But we think that it is even more important to have checked that binarization helps to effectively apply conventional diversification mechanisms, as well as to have experimentally demonstrated that combining other techniques with them (namely pre-emphasis and data augmentation) further increases the ensemble performance.

4. Pre-emphasis and data augmentation

4.1. The concept of pre-emphasis and our selected formulations for it

Training sample selection techniques and their evolution, training sample weighting, called also pre-emphasis, are efficient and effective methods to improve the performance of classification machines. Conceptually, they are mechanisms that indicate to the machine the degree of attention which must be paid to each sample, by means of weighting the corresponding training cost component. If weights are restricted to {0, 1}, we have a sample selection scheme, which was the form applied in the seminal work of Hart with nearest neighbor classifiers fifty years ago [66]. Many other forms have been subsequently proposed, among which [67–70] include interesting alternatives and discussions. Pre-emphasis methods have recently been used for selecting samples as kernel centroids [71] and also for allowing a direct training of Gaussian Process classifiers by defining an appropriate form of soft targets [72].

The weight that each sample receives has been related to its classification error or to a measure of its proximity to the classification boundary, usually according to the output of an auxiliary classifier. The effectiveness of both methods is problem-dependent [73]. Therefore, the pre-emphasis forms that are applied today include components of the two kinds. These combinations have demonstrated their effectiveness for designing boosting ensembles [42,43].

In this paper, we will adopt the general forms of weighting functions that we have previously applied just to pre-emphasize SDAE-3 classifiers with excellent results [74,75]. For binary problems, the weight for the training cost of the sample $\{\mathbf{x}^{(n)}, t^{(n)}\}$, $t^{(n)} \in \{-1, 1\}$, is

$$p(\mathbf{x}^{(n)}) = \alpha + (1-\alpha)[\beta(t^{(n)} - o_a^{(n)})^2 + (1-\beta)(1 - o_a^{(n)2})] \quad (1)$$

where $o_a^{(n)}$ is the output of the auxiliary classifier when $\mathbf{x}^{(n)}$ is its input, and the parameters α, β, $0 \leq \alpha, \beta \leq 1$, serve to establish the appropriate proportion of no emphasis (the term α), error emphasis (the term $(1-\alpha)\beta(t^{(n)} - o_a^{(n)})^2$), and proximity emphasis (the term $(1-\alpha)(1-\beta)(1 - o_a^{(n)2})$). In general, values for α, β can be found by means of cross-validation processes. Note that form (1) includes many particular cases: $\alpha = 0$, full emphasis with the two components; $\beta = 0$, moderated (by α) emphasis according to the proximity to the boundary; $\beta = 1$, moderated emphasis according to the error; $\alpha = 0$ and $\beta = 0$, full emphasis according to the proximity to the boundary; $\alpha = 0$ and $\beta = 1$, full emphasis according to the error; and $\alpha = 1$, no emphasis at all. Obviously, there are other possible functional forms for the error and the proximity to the boundary terms. The different forms are more or less effective in a problem-dependent manner, but the performance differences are always moderate.

For the (softmax) multi-class machines we will use

$$p(\mathbf{x}^{(n)}) = \alpha + (1-\alpha)[\beta(1 - o_{ac}^{(n)})^2 + (1-\beta)(1 - |o_{ac}^{(n)} - o_{ac'}^{(n)}|)] \quad (2)$$

where $o_{ac}^{(n)}$ is the output of the auxiliary (softmax) machine corresponding to the correct class c for $\mathbf{x}^{(n)}$, and $o_{ac'}^{(n)}$ the output of that machine whose value is the nearest to $o_{ac}^{(n)}$ among the rest of classes.

We remark that the application of pre-emphasis demands an additional computational effort in designing classifiers, due to the need of validation to find appropriate values for α, β. But there is no increase in the operation –i.e., the classification of unseen samples– computational effort, since the classifier architecture remains the same. This was the reason which moved us to include pre-emphasis in a second step of experiments, after checking that the results offered by binarization and conventional diversity were satisfactory.

4.2. Data augmentation

Data augmentation methods pre-process training examples to create new samples having similar characteristics to those of the original ones. They allow to increase the number of training samples by adding the augmented versions with labels corresponding to the original samples from which they have been obtained. If carefully selected and designed, these methods are effective in order to increase the performance of classification machines.

Data augmentation has been used along two decades [76], and its forms have significantly evolved. We repeat that the MNIST classification absolute record was obtained by means of a CNN ensemble with data augmentation as the diversification source [29]. There are several data augmentation mechanisms that have shown effectiveness in improving the performance of image classifiers, such as random translations, random rotations, centering, and elastic deformations. For concise reviews, we recommend [28,77]. Given the advantage that data augmentation provides and that, as for pre-emphasis, there is not an increase of the operation computational effort, we also include a form of it, the most frequent version of elastic deformation [78], in the last step of our experiments, after combining pre-emphasis and diversity.

The basic aspects of the elastic deformation we will employ are as follows. First, there is a pixel translation, whose horizontal and vertical values are obtained by multiplying the elements of two matrices of the image size with small random values by a scale factor, Δ, which must be appropriately selected. Translations are limited to the image borders, and values that arrive to the same position are averaged. After it, a normalized Gaussian filtering is carried out with the objective of smoothing the results. The parameter σ of the filter must also be selected with care. Our selection of Δ, σ, will be discussed in the section dedicated to the experiments.

5. Experimental framework

We concentrate here the general aspects of our experiments to avoid disorienting the readers with a disperse presentation.

5.1. Dataset

As previously announced, we will work with a much studied, relatively easy dataset of handwritten digits, MNIST [3], both because there are many published results for different machine classifiers and because we want to see if the techniques we propose in this paper produce enough improvements as to get competitive results with those of CNN designs when a general propose basic deep networks, SDAE-3, is used. MNIST contains 50,000 / 10,000 / 10,000 training / validation / test samples, respectively, of dimension 784 (28×28) with 256-level quantized values normalized into [0,1].

It seems important to analyze the effect of the strongness of the learners when building an ensemble. Fortunately, there is a version of MNIST, MNIST-BASIC [9] (MNIST-B in the rest of this paper), whose only difference with MNIST is that its training / validation / test samples are 10,000 / 2000 / 50,000, respectively. Less training samples will imply weaker learners, and, since MNIST and MNIST-B have the same nature, this will permit an excellent perspective for the analysis we mention.

Finally, we include also a binary database because it is very relevant to appreciate the possible differences between multi-class and binary problems in our study. For reasons of similarity, we selected RECTANGLES (RECT in the following), a database with the outlines of white horizontal or vertical rectangles on a black background of just 28×28 pixels. The sizes of the training / validation / test sets are 10,000 / 2000 / 50,000, respectively.

5.2. The basic deep machine: the SDAE-3 classifier

Once again, we emphasize that we select a DNN without a convolutional structure because we want to check if the procedures we are introducing are effective enough to lead to high performance results using a general propose architecture, and CNNs can be considered "ad hoc" architectures when dealing with images – our datasets in this work– and other translational inputs, such as speech. Under that condition, there is no reason for expecting that things will be different, in general, for other kinds of problems and DNNs.

We have selected the 3-layer Stacked Denoising Auto-Encoder (SDAE-3) classifier [9] both because it is a representation DNN and because it has been applied to the databases we are working with. Consequently, we can adopt the architecture and training parameters of [9] for a better appreciation of the effects of our proposed techniques.

Fig. 1 shows the structure of an SDAE-3 classifier. It consist of three expansive auto-encoding layers with sigmoidal activations, plus a final classification unit, CL, with a sigmoidal or a softmax output for binary or multi-class problems, respectively. The auto-encoding layers are sequentially trained using the input samples $\mathbf{x}^{(n)}$ as targets for noisy input vectors $\mathbf{x}^{(n)} + \mathbf{r}$. The denoising makes possible the expansive architecture, which increases the expressive capacity, and also provides some degree of robustness. Each auto-encoding hidden layer is frozen before training the next. Finally, the top classifier is trained and the auto-encoding weights refined.

The design parameters used in [9] were 1000 units for the auto-encoding layers, an 1,000-hidden layer MLP final classifier, a sample-by-sample BP training with a 0.01 learning step for the first auto-encoding layer and 0.02 for the rest and the top MLP and refining, and 40 training epochs, that are enough for convergence. We checked these values with positive results. However, our results were better with an added noise variance 10% the variance of the samples, leading us to the classification results (for 10 different initialization runs), % average error rates $\pm$ standard deviation, of 1.58 $\pm$ 0.06, 3.42 $\pm$ 0.10, and 2.40 $\pm$ 0.10 for MNIST, MNIST-B, and RECT, respectively. These results are slightly worse than those of [9], but the standard deviations are lower.

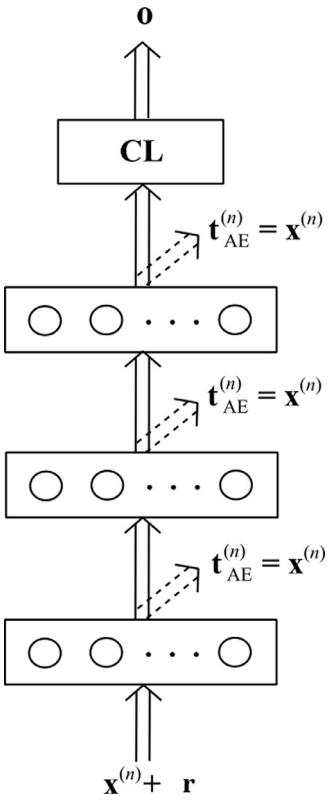

Fig. 1. SDAE-3 classifier (based on stacked denoising auto-encoders). $\mathbf{x}^{(n)}$: training sample; $\mathbf{r}$: training noise. The weights of the layers are consecutively obtained by imposing the noisy-free input samples $\mathbf{x}^{(n)}$ as targets, and then they are frozen until inserting the final classifier, CL. $\mathbf{o}$ is the overall output.

5.3. Conventional diversification

We use bagging and switching. The determination of the sizes of the corresponding ensembles is done according to the best results for the validation set, using $N = 25, 51, 101,$ for the number of ensemble learners. The same validation set serves to select the bootstrapping size among $B = 60, 80, 100$ and 120% that of the original train set, and the switching rate among $S = 10, 20, 30$ and 40% of the training samples. All committee's units are initialized with uniformly distributed random values.

5.4. Binarization techniques

According to the discussion of Section 2.2, we will apply OvO binarization and the 15 bit, 10-class ECOC which is proposed in [51] just for handwritten digits.

6. Experimental results and their discussions

6.1. First level experiments: binarization and conventional diversity

We repeat that our preliminary experiments with multi-class problems MNIST and MNIST-B using only bagging or switching of SDAE-3 classifiers did not provide any performance improvement, the same negative result that other studies obtained. Thus, we carried out additional experiments including both binarization and bagging or switching.

Two alternative approaches were considered. The first consists of diversifying the SDAE-3 auto-encoding parts, that we will indi-

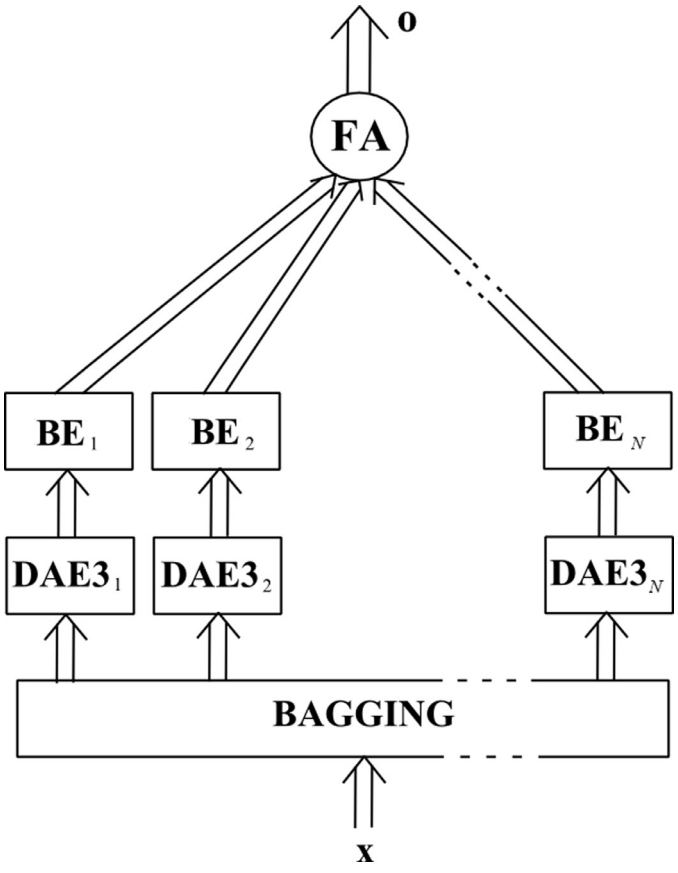

Fig. 2. The G model for multiclass problems. DAE3$_n$ are deep expansive denoising auto-encoders, BE$_n$ the binarizing ensembles, and FA the final aggregation (see text for details). **o** is the overall output.

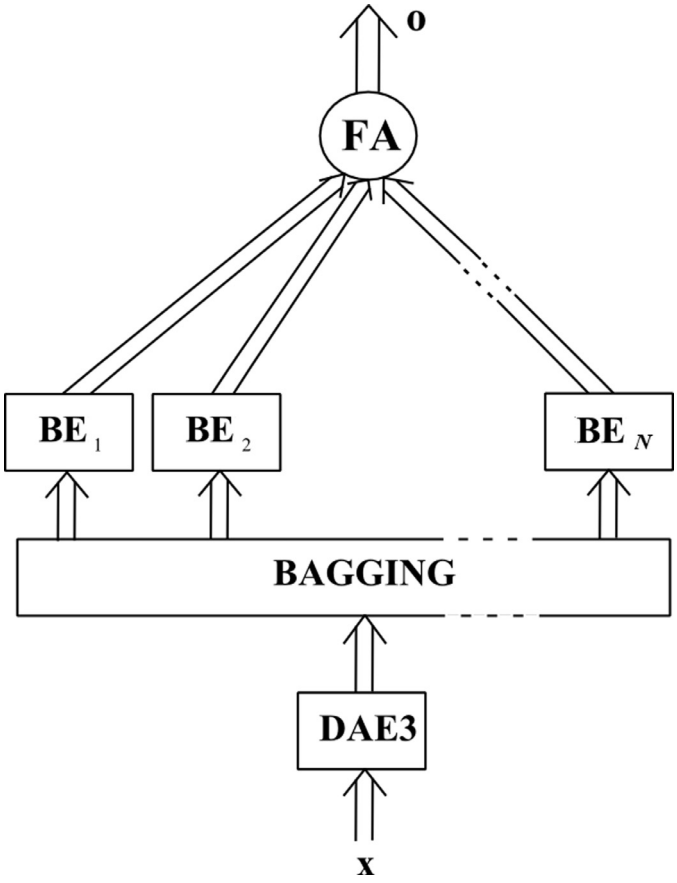

Fig. 3. The TB model for multiclass problems (for bagging). DAE3 is a single expansive denoising deep auto-encoder, BE$_n$ the binarizing ensembles, and FA the final aggregation (see text for details). **o** is the overall output.

cate as DAE3, and, after it, for multi-class problems, to construct binarization ensembles with final classifiers for each DAE3 output. Fig. 2 shows the corresponding model. The final aggregation, FA, is a vote counting for each ensemble plus a majority vote if OvO is applied, or a vote counting plus a Hamming distance class selection for ECOC binarization. Obviously, switching cannot be applied to build DAE3 ensembles. We will call G (Global) model to this form of constructing ensembles and binarizing.

Fig. 3 represents the second alternative, which we call the T model because its aspect: There is a unique DAE3, bagging or switching are applied at its output, and then, OvO or ECOC binarizing ensembles of final classifiers are trained with the diverse training sets. This model is suggested by the experiences indicating that diversification at higher layers is more effective, such as those presented in [61]. The final aggregation is identical to that of the G model.

As we announced in Section 5.3, the values of the non-trainable parameters are selected according to the results for the validation sets among $N = 25, 51,$ and 101 for the ensemble size, $B = 60, 80, 100,$ and 120% for the bootstrap training set sizes, and $S = 10, 20, 30,$ and 40% for the switching rates. No refining is carried out when training.

To reduce the huge computational load for the design of G models, we apply a frequently used simplification: We build $M > N$ bagging units for each value of the bootstrap sample sizes, and then take N at random. Our experience with this trick indicates that the performance degradation (due to a loss in diversity) is very moderate.

Table 1
Part a: performance results (% average error rate ± typical deviation) for OvO binarization of multi-class problems. DAE3: single DAE with OvO binarization; GB: G model with bagging; TB: T model with bagging; TS: T model with switching. Part b: performance results for the model T with switching when replacing OvO by ECOC binarization. Validation selected non-trainable parameter values are also indicated. Of course, RECT designs do not include binarization. Best results appear in boldface.

	MNIST	MNIST-B	RECT (no binarization)
SDAE-3	1.58 ± 0.06	3.42 ± 0.10	2.40 ± 0.13
a			
DAE3, OvO	1.40 ± 0.06	2.60 ± 0.08	–
GB, OvO	0.86 ± 0.01	1.76 ± 0.04	1.20 ± 0.04
(N, B)	(101, 120)	(101, 120)	(101, 120)
TB, OvO	0.77 ± 0.00	1.68 ± 0.04	1.19 ± 0.01
(N, B)	(101, 100)	(101, 120)	(101, 120)
TS, OvO	0.75 ± 0.00	1.67 ± 0.04	**1.10 ± 0.02**
(N, S)	(101, 40)	(101, 40)	(101, 40)
b			
TS, ECOC	**0.36 ± 0.02**	**0.75 ± 0.01**	–
(N, S)	(101, 30)	(101, 30)	

6.1.1. Results for OvO binarization and their discussion

Part a of Table 1 shows the performance results for the G and T architectures with N and B/S values selected by validation and for a DAE3 plus an OvO binarization (without conventional diversification).

Before discussing these performance results, a few words about the selected values for non-trainable parameters. In all the cases but one (model T with bagging), these selected values are the high-

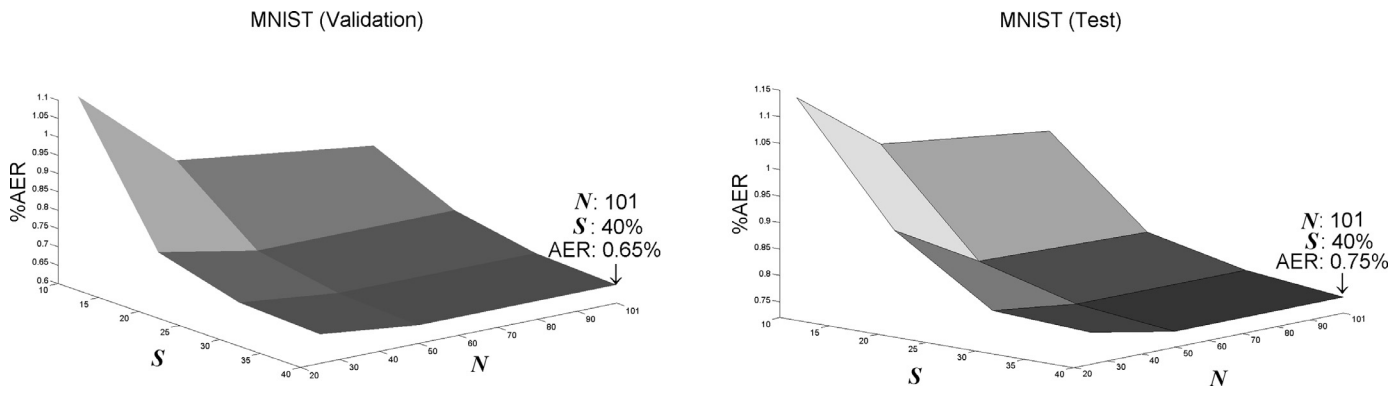

Fig. 4. The average error rate for the validation and test sets versus N and S for the MNIST data set using the T model and OvO binarization.

est among those explored. But analysis of the performance for different pairs of values makes evident that there is a clear saturation effect just appearing for these extreme values ($N = 101, B = 120\%, S = 40\%$). This can be seen for the MNIST database in Fig. 4. Therefore, the best is just to adopt these values, because increasing them will only increase the computational load to classify unseen samples, but not the performance. It must also be remarked that the parallelism between the performance surfaces for the validation and the test sets is nearly perfect. This means that the validation process will provide good values for the non-trainable parameters.

From the performance results in Part a of Table 1 it can be concluded that bagging and switching diversification are effective if combined with binarization when dealing with multi-class problems (MNIST and MNIST-B), and that binarization by itself is not the main reason for these improvements, because binarizing the classifiers of a simple DAE3 only provides modest performance increases. Thus, it appears that conventional diversification with powerful classifiers when dealing with multi-class problems is not able of improving the corresponding classification boundaries due to the complexity of these boundaries.

It is also evident that performance improvements are more important for model T than for model G approaches. This seems to indicate that the auto-encoding layers of the SDAE-3 classifiers carry out their disentangling function in an effective and efficient manner. Consequently, model T designs must be preferred, because their computational design and operation costs are lower. On the other hand, there are not significant differences between bagging and switching model T results.

To close this discussion, let us remark that the performance increases in an important amount: The average error rates for the switching model T cases are 47%, 49%, and 46% of those of single SDAE-3 classifiers for MNIST, MNIST-B, and RECT, respectively, and typical deviations become lower. Note that in this case there are not qualitative differences between the multi-class problems and RECT, nor between MNIST and MNIST-B ('strong' and 'weak' learners).

6.1.2. Results for the ECOC binarization and their discussion

According to the conclusions of the OvO binarization experiments, we will restrict our ECOC design to model T forms and just one of the conventional diversification techniques, switching. The explored non-trainable parameter values are $N = 21, 51, 101,$ and 121, and $S = 10, 20, 30,$ and 40%.

Part b of Table 1 shows the experimental results. Performance figures are much better than those with OvO binarization, which confirms the advantage of well-designed ECOC binarization procedures. An average error rate of 0.36% for MNIST is a very good result using non-convolutional deep machines.

Of course, all these improvements do not come by free: Computational costs are much more important than those of a single SDAE-3, both for training and for classifying unseen samples, or operation. Considering the ECOC T case for the MNIST problem, a direct count gives a total of around 1.5×10^9 weights and around 1.5×10^6 sigmoids, that are approximately 4×10^2 times the numbers for a single SDAE-3 classifier. This is a reasonable estimate for the operation computational load increase. And a detailed accounting of the operations per training step for the diversified and binarized final classifier compared with those required for a single SDAE-3 gives a similar order of magnitude for the load increase: Even considering that these classifiers will become trained in less steps, this is also a large computational effort. But, as when dealing with shallow machines, this is the price to be paid to get advantage from binarization and conventional diversification. Obviously, to justify their application, the problem to be solved has to show a high overall misclassification cost.

6.2. Including pre-emphasis

As an introduction, we will resume the results of applying only pre-emphasis to SDAE-3 classifiers [74,75] before adding this technique to binarization and conventional diversity. This will serve to appreciate if the combination is better than all of its components.

6.2.1. Pre-emphasizing SDAE-3 classifiers

We adopt the same SDAE-3 classifier architecture and parameters that above and we apply multi-class or binary pre-emphasis formulas (1), (2). The parameters α, β, are explored in 0.1 steps along the interval [0, 1].

With respect to the auxiliary classifiers, or guides, our experience working with shallow classifiers is that pre-emphasis effects are better when using better guides. This can be expected, because better auxiliary classifiers provide better classification results, and, consequently, more appropriate pre-emphasis weights. Additionally, pre-emphasis effects are also better if the guide architecture is similar to that of the pre-emphasized machine. This is also reasonable, because both classifiers are able of constructing similar boundaries, and this fact implies that more benefit can be obtained from the pre-emphasis process. However, we considered appropriate to check if these findings can be extended to deep classifiers. So, we applied two kinds of auxiliary machines: One-hidden layer with 1000 units MLPs, and the single SDAE-3 classifier without pre-emphasis.

On the other hand, when pre-emphasizing SDAE-3 classifiers, it is unclear if it will be better to apply the pre-emphasis to both the auto-encoding layers and the final classification or only to the final classification step. We tried both alternatives, and here will indicate them as "initial" and "final" designs.

Table 2
Test error rate in percent plus or minus the standard deviation for the three databases using pre-emphasized SDAE-3 classifiers. The values for α, β obtained by validation are given in parentheses. A star (*) indicates suboptimal results (see the text for a discussion). Best results appear in boldface.

Pre-emphasis	Aux. classifier	MNIST	MNIST-B	RECT
None (MLP)	–	2.66 ± 0.10	4.44 ± 0.23	7.20 ± 0.15
None (SDAE-3)	–	1.58 ± 0.06	3.42 ± 0.10	2.40 ± 0.13
Initial	MLP	0.40 ± 0.04* (0.3, 0.6)	0.82 ± 0.01 (0.3, 0.5)	0.92 ± 0.10 (0.4, 0.4)
Initial	SDAE-3	**0.37 ± 0.01*** (0.4, 0.5)	**0.72 ± 0.01** (0.3, 0.5)	**0.87 ± 0.04** (0.4, 0.3)
Final	MLP	0.57 ± 0.00 (0.4, 0.6)	0.91 ± 0.03 (0.3, 0.5)	1.26 ± 0.04 (0.6, 0.3)
Final	SDAE-3	0.67 ± 0.05* (0.4, 0.4)	0.83 ± 0.02 (0.3, 0.5)	1.31 ± 0.02* (0.4, 0.3)

Table 2 shows the results of the corresponding experiments.

It can be seen that any of the four pre-emphasis mechanisms improves the performance of the conventional SDAE-3 classifiers, but also that the initial pre-emphasis with the SDAE-3 classifier guide is clearly the best option. This confirms our expectatives, and also reveals that it is important to apply the pre-emphasis even to the auto-encoding process. This result can be easily explained: A better representation of the training samples that are more important to define the classification boundaries produces performance benefits.

As in the diversification process, there appears a clear parallelism between the performance surfaces vs. α, β, for the validation and the test sets. However, these surfaces are not so smooth, and this produces some sub-optimal results –other pairs of α, β values would offer better test performance,– that are indicated by asterisks. Differences are very minor, the most relevant case being the design with the SDAE-3 classifier guide and initial emphasis for MNIST: $\alpha = 0.3$, $\beta = 0.5$ would give a 0.36% average error rate with a (practically) zero standard deviation. Clearly, a very minor change with respect to the validated result.

We can say that results are very good. In particular. 0.37, 0.72, and 0.87% average error rates for MNIST, MNIST-B, and RECT, respectively, with a single SDAE-3 classifier are excellent: It must be considered that the operation computational load does not increase, and that the design requires only $11 \times 11 = 121$ times that of a conventional single SDAE-3 classifier, due to the validation of α, β. Note that the above error rates are 0.23, 0.21, and 0.36% those of the single SDAE-3 classifiers without pre-emphasis for MNIST, MNIST-B, and RECT, respectively. The higher improvements for the multi-class problems are easy to understand: Helping to find better classification boundaries is more important when there are multiple boundaries.

Let us remark the critical importance of using general and flexible enough weighting formulas such as (1) and (2): All the validation selected pairs have values of α, β far from 0 and 1, that correspond to more limited emphasis forms. To show the degradation that limited weighting schemes produce, it must suffice to list their performance for the MNIST dataset with an SDAE-3 classifier guide and initial emphasis (% average error rates):

- $\alpha = 1$ (no emphasis): 1.58 (that of the conventional SDAE-3 classifier)
- $\alpha = 0$, $\beta = 1$ (full error emphasis): 0.85
- $\alpha = 0$, $\beta = 0$ (full proximity emphasis): 0.71
- $\alpha = 0$, $\beta = 0.6$ (full combined emphasis): 0.58
- $\alpha = 0.2$, $\beta = 1$ (moderated error emphasis): 0.77
- $\alpha = 0.2$, $\beta = 0$ (moderated proximity emphasis): 0.58

Table 3
Performance results (% average error rate ± standard deviation) for the two ensembles that are trained with initial pre-emphasis. PrE T(ECOC) S corresponds to the best ensemble of Section 6.1, and PrE ECOC S corresponds to an ensemble whose first diversity is the ECOC binarization. The validated values of α, β are also shown for the first design. Best results appear in boldface.

	MNIST	MNIST-B	RECT (no binarization)
PrE T(ECOC)S	0.30 ± 0.01 (α = 0.2, $\beta = 0.4$)	0.62 ± 0.01 (α = 0.2, $\beta = 0.6$)	**0.76 ± 0.02** ($\alpha = 0.4$, β = 0.3)
PrE ECOC S	**0.26 ± 0.04**	**0.55 ± 0.04**	–

All the above performances are clearly worse than the average error rate which $\alpha = 0.4$, $\beta = 0.5$ offers, 0.37%.

6.2.2. Pre-emphasizing binarized and diversified SDAE-3 classifiers

For the sake of brevity, we will present here the results of applying initial pre-emphasis with the conventional SDAE-3 classifier guide only for the best ensembles of those presented in Section 6.1, TS with ECOC binarization (simply TS for RECT) –results for the other cases are worse,– and, in the cases of multi-class problems, for an alternative which permits to apply a different pre-emphasis for each ECOC binary problem: First, the ECOC is applied, and then the DAE3 auto-encoders plus the final switching ensembles complete each branch, and values of α, β are separately selected for those branches. The final aggregation is the same voting plus Hamming distance based selection. We will also keep the non-trainable parameters previously used, since there is not a significant sensitivity with respect to them. We will indicate these pre-emphasized ensembles as PrE T(ECOC)S and PrE ECOC-S, respectively.

The experimental results appear in Table 3.

Comparing the first row of these results with those of the last row of Table 1 and the fourth row of Table 2 makes evident that combining pre-emphasis and diversity (including binarization for multi-class problems) produces significant improvements in performance. And the results that appear in the last row of Table 3 indicate that separate pre-emphasis is even more effective to increase the classification performance. Obviously, separate pre-emphases require more design computational effort, because their α, β parameters must be independently selected by means of the corresponding validation processes. However, the operation computational load for both pre-emphasized ensembles is of the same order of magnitude, around 1.5×10^9 multiplications.

Once more, we insist: Such a huge computational effort is the price to be paid to get the exceptional performances of these ensembles of SDAE-3 classification machines in the image problems we consider in our experiments, that are, for the PrE ECOC-S model, 16% and 16% the error rates of a conventional single SDAE-3 classifier for MNIST and MNIST-B, respectively.

Although the above results are excellent –note that we get an error rate for MNIST which is only 1.24 times the absolute record, which was reached with a CNN ensemble [29],– the question of what are the limits of these kind of approaches with general purpose, computationally expensive DNNs emerges. Direct attempts of improving the performance, such as a second validation round with a finer grid of values for α, β are not the solution: A two digit second round search for the PrE ECOC S ensemble leads to "only" 0.24 ± 0.08 and 0.52 ± 0.06 average error rates ± standard deviation for MNIST and MNIST-B, respectively. Yet the answer is to check if other improvement mechanisms can be combined with those we have already applied, as the success of adding pre-emphasis to binarization and conventional diversification suggests.

Table 4
% error average rate ± standard deviations for the PrE ECOC S ensemble when training samples are augmented by using the Elastic Distortion (ED) process described in the main text.

	MNIST	MNIST-B
ED PrE ECOC S	0.19 ± 0.01	0.50 ± 0.03

In the next section, we will explore the additional application of a data augmentation technique, elastic deformations.

7. Adding elastic deformations

Data augmentation has been a traditional complement of handwritten digit classification algorithms [76], leading to very good performances [28], in particular when applied for building ensembles [29]. Therefore, to add it to the above designs is an interesting possibility.

We presented an overview of the procedure we will apply here, an elastic distortion [78], in Section 4.2. We work with diverse distortions, selecting five combinations of parameters that produce visually acceptable results: $\{\Delta, \sigma\} = \{3, 30\}, \{4, 20\}, \{4, 30\}, \{5, 10\}$, and {5, 20}, where the scale parameter are applied to horizontal and vertical displacement matrices that are formed by $[-0.1, 0.1]$ uniform random values. We explore the appropriate generation rate (of distorted samples with respect to original examples) among 100, 200, 300, and 400% the number of training samples, finding that 300% is good for most of the designs. Since we are including distorted random samples, we also explore the added noise level around the 10% value which was previously applied, selecting 7% as the appropriate variance.

Keeping the rest of the non-trainable parameters at their values, except α, β, that are validated as before, we obtain the % error rates ± standard deviations of Table 4 when adding the above elastic distortions for training the PrE ECOC S designs. As before, 10 runs are considered.

It can be seen that there are significant further performance improvements, and this supports the intuition that combining the elastic distortion with the previously applied pre-emphasis plus binarization and conventional diversification, is effective. We remark that the performance for MNIST is a new absolute record, superior to the performance of the CNN ensembles of [29], even CNN being more adapted machines for the handwritten digit recognition task. These facts permit to conjecture that combining diversity with other improvement mechanisms having different natures will very likely provide performance advantages when working with DNNs.

To appreciate the effectiveness of our record design, Fig. 5 shows the wrongly classified digits for a typical run. It is easy to see that these are difficult samples even for a human expert.

Once again, a high increase of computational costs is the price to be paid in order to obtain these important performance improvements. In particular, the record design has an architecture that differs from the ECOC T considered in Section 6.1.2 only in including 15 SDAE-3, one for each ECOC binary problem. Since the multiplications and sigmoidal transformations in these SDAE-3 are two orders of magnitude lower than those of the conventionally diversified final classifiers, we have again around 1.5×10^9 multiplications and 1.5×10^6 sigmoidal transformations for the operation computational cost. We remark that both pre-emphasis and elastic deformations increase the design effort, but they do not modify the size of the designed machine and, consequently, they do not change the operation computational load.

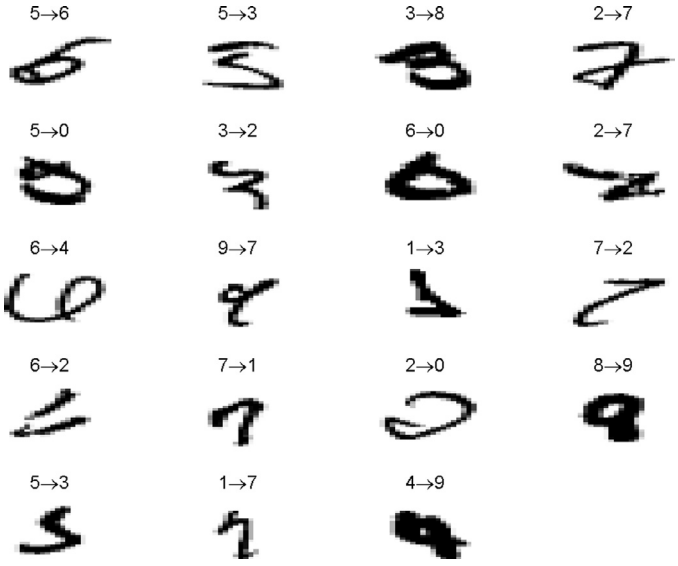

Fig. 5. Misclassified MNIST digits in a typical run of the ED PrE ECOC S design. k→ l: true class→ classification result.

8. Conclusions and further work

Applying a general-purpose representation basic DNN, the SDAE-3 classifier, to a selected numbers of datasets (MNIST, MNIST-BASIC, and RECTANGLES) that are relatively simple but that we choose to allow appreciating the effects of their different characteristics, we have found the following experimental facts:

- Building diverse ensembles (by means of bagging and switching, in particular) is efficient to increase classification performance, but multi-class problems require the simultaneous application of binarization techniques, that, by themselves, only produce modest advantages.
- Conventional diversification is more effective when applied at the last classification steps.
- Applying pre-emphasis sample weighting is also effective, in particular when the weighting formulas are general and flexible. These general and flexible forms produce very important performance improvements, without increasing the computational load to classify unseen samples.
- Combining pre-emphasis with binarization and conventional diversity further improves the performance results, in a very remarkable manner when a different pre-emphasis is applied to each binary problem in multi-class situations.
- Adding to the above combination an elastic distortion process to create appropriate additional training samples produces once more an increased performance, by no means trivial: This way has lead us to a new absolute record in classifying MNIST digits.
- All the above techniques are useful when they are combined, if the appropriate forms are selected.

Of course, much more work is necessary to check what is the advantage that these methods and their combinations provide when addressing other classification problems and/or using other DNN classifiers, and we are actively working in this direction. We advise that CNN is a delicate architecture which opposes serious difficulties to conventional methods of building ensembles, but, even if this approach is not successful, there are many additional possibilities that can be explored to improve its performance –and that of other DNN,– and it is plausible that combining them will permit higher performance advantage, at least if the combined procedures have different character, i.e., different conceptual reason to produce performance improvements. And let us

clarify that we are speaking non only of pre-emphasis or elastic distortion, but also of other data augmentation techniques, more elaborated methods of noisy learning (considering the problem to be addressed), and regularization mechanisms, including drop-out or drop-connect.

Note

In http://www.tsc.uc3m.es/ ~ralvear/Software.htm, the interested reader can access the software blocks we have developed for our experiments, as well as find links to other blocks we have used.

Acknowledgements

This work has been partly supported by research grants CASI-CAM-CM (S2013/ICE-2845, Madrid Community) and Macro-ADOBE (TEC2015-67719, MINECO-FEDER EU), as well as by the research network DAMA (TIN2015-70308-REDT, MINECO).

We gratefully acknowledge the help of Prof. Monson H. Hayes in discussing and preparing this work.

References

[1] G. Cybenko, Approximation by superpositions of a sigmoidal function, Math. Control Signals Syst. 2 (1989) 303–314.

[2] K. Hornik, M. Stinchcombe, H. White, Multilayer feedforward networks are universal approximators, Neural Netw. 2 (1989) 359–366.

[3] Y. LeCun, B. Boser, J.S. Denker, D. Henderson, R.E. Howard, W. Hubbard,
L.D. Jackel, Backpropagation applied to handwritten zip code recognition, Neural Comput. 1 (1989) 541–551.

[4] Y. Bengio, Learning deep architectures for AI, Found. Trends Mach. Learn. 2 (2009) 1–127.

[5] J. Schmidhuber, Deep Learning in Neural Networks: An Overview, Technical report, IDSIA-03-14, University of Lugano, 2014. arXiv: 1404.7828v4 [cs.NE].

[6] G.E. Hinton, S. Osindero, Y. Teh, A fast learning algorithm for deep belief networks, Neural Comput. 18 (2006) 1527–1554.

[7] P. Smolensky, Information processing in dynamical systems: foundations of harmony theory, in: D.E. Rumelhart, J.L. McClelland, The PDP Research Group (Eds.), Parallel Distributed Processing: Exploration in the Microstructure of Cognition. Vol. 1: Foundations, Cambridge, MA: MIT Press, 1986, pp. 194–281.

[8] M.A.C.-P. Carreira-Perpiñán, G.E. Hinton, On contrastive divergence learning, in: Proceedings of the 10th International Workshop on Artificial Intelligence and Statistics, Barbados: Soc. Artificial Intelligence and Statistics, 2005, pp. 33–40.

[9] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, P.A. Manzagol, Stacked denoising autoencoders: learning useful representations in a deep network with a local denoising criterion, J. Mach. Learn. Res. 11 (2010) 3371–3408.

[10] Y. Bengio, A. Courville, P. Vincent, Representation learning: a review and new perspectives, IEEE Trans. Pattern Anal. Mach.Intell. 35 (2013) 1798–1828.

[11] M.M. Najafabadi, et al., Deep learning application and challenges in big data analytics, J. Big Data 2 (1) (2015) 9.

[12] P.P. Brahman, D. Wu, Y. She, Why deep learning works: a manifold disentanglement perspective, IEEE Trans. Neural Netw. Learn. Syst. 27 (2016) 1997–2008.

[13] L. Deng, D. Yu, Deep convex net: a scalable architecture for speech pattern classification, in: Proceedings of Interspeech 2011, Florence, Italy, 2011, pp. 2285–2288.

[14] B. Hutchinson, L. Deng, D. Yu, A deep architecture with bilinear modeling of hidden representations: applications to phonetic recognition, in: Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing, New York, NY: IEEE Press, 2012, pp. 4805–4808.

[15] X. Glorot, Y. Bengio, Understanding the difficulty of training deep feedforward neural networks, in: Proceedings of the 13th International Conference on Ar- tificial Intelligence and Statistics, JMLR: W&CP 9, Chia Laguna Resort, Sardinia, Italy, 2010, pp. 249–256.

[16] J. Martens, Deep learning via Hessian-free optimization, in: Proceedings of the 23th International Conference on Machine Learning, Haifa (Israel), 2010, pp. 735–742.

[17] J. Duchi, E. Hazan, Y. Singer, Adaptive subgradient methods for online learning and stochastic optimization, J. Mach. Learn. Res. 12 (2011) 2121–2159.

[18] V. Nair, G.E. Hinton, Rectifier linear units improve resticted Boltzmann ma- chines, in: Proceedings of the 23th International Conference on Machine Learning, Haifa (Israel), 2010, pp. 807–814.

[19] S. Ioffe, C. Szegedy, Batch normalization: accelerating deep network training by
reducing internal covariate shift, Proceedings of the 32th International Conference on Machine Learning, JMLR: W&CP28. Lille (France), 2015.

[20] I. Sutskever, G.E. Hinton, Deep, narrow sigmoid belief networks are universal approximators, Neural Comput. 20 (2008) 2629–2636.

[21] G. Montufar, N. Ay, Refinements of universal approximation results for deep belief networks and restricted Boltzmann machines, Neural Comput. 23 (2011) 1306–1319.

[22] L. Szymanski, B. McCane, Deep networks are effective encoders of periodicity, IEEE Trans. Neural Netw. Learn. Syst. 25 (2014) 1816–1827.

[23] J. Håstad, M. Goldmann, On the power of small-depth threshold circuits, Comput. Complexity 1 (1991) 113–129.

[24] O. Delalleau, Y. Bengio, Shallow vs. deep sum-product networks, in: J. Shawe–Taylor, R.S. Zemel, P.L. Bartlett, F. Pereira, K.Q. Weinberger (Eds.), Advances in Neural Information Processing Systems 24, Cambridge, MA: MIT Press, 2011, pp. 666–674.

[25] L. Deng, D. Yu, Deep learning: methods and applications, Found. TrendsSignal Process. 7 (2014) 197–387.

[26] Deep Learning University, An annotated deep learning bibliography, http:// memkite.com/deep-learning-bibliography/.

[27] D. Miskin, J. Matas, All You Need is a Good Init, Technical report, IDSIA-03-14, University of Lugano, 2015. arXiv: 1511.06422v7 [cs.LG].

[28] S. Tabik, D. Peralta, A. Herrera-Poyatos, F. Herrera, A snapshot of image pre-processing for convolutional neural networks: case study of MNIST, Int. J. Comput. Intell. Syst. 10 (2017) 555–568.

[29] D. Ciresan, U. Meier, J. Schmidhuber, Multi-column deep neural networks for image classification, in: Proceedings of the Conference on Computer Vision and Pattern Recognition, New York, NY: IEEE Press, 2012, pp. 3642–3649.

[30] C.K. Chow, Statistical independence and threshold functions, IEEE Trans. Electron. Comput. 14 (1965) 66–68.

[31] A.J. Sharkey, Combining Artificial Neural Nets: Ensemble and Modular Multi--Net Systems, London, UK: Springer, 1999.

[32] L.I. Kuncheva, Combining Pattern Classifiers: Methods and Algorithms, Hoboken, NJ: Wiley, 2004.

[33] L. Rokach, Pattern Classification Using Ensemble Methods, Singapore: World Scientific, 2010.

[34] R.E. Schapire, Y. Freund, Boosting: Foundations and Algorithms, Cambridge, MA: MIT Press, 2012.

[35] M. Woźniak, M.G. na, E. Corchado, A survey of multiple classifier systems as hybrid systems, Inf. Fusion 16 (2014) 3–17.

[36] L. Breiman, Random forests, Mach. Learn. 45 (2001) 5–32.

[37] L. Breiman, Bagging predictors, Mach. Learn. 24 (1996) 123–140.

[38] L. Breiman, Randomizing outputs to increase prediction accuracy, Mach. Learn. 40 (2000) 229–242.

[39] G. Martínez-Muñoz, A. Sánchez-Martínez, D. Hernández-Lobato, A. Suárez, Class-switching neural network ensembles, Neurocomputing 71 (2008) 2521–2528.

[40] Y. Freund, R.E. Schapire, A decision-theoretic generalization of on-line learning and an application to boosting, J. Comput. Syst. Sci. 55 (1997) 119–139.

[41] R.E. Schapire, Y. Singer, Improved boosting algorithms using confidence-rated predictions, Mach. Learn. 37 (1999) 297–336.

[42] V. Gómez-Verdejo, M. Ortega-Moral, J. Arenas-García, A.R. Figueiras-Vidal, Boosting by weighting critical and erroneous samples, Neurocomputing 69 (2006) 679–685.

[43] V. Gómez-Verdejo, J. Arenas-García, A.R. Figueiras-Vidal, A dynamically adjusted mixed emphasis method for building boosting ensembles, IEEE Trans. Neural Netw. 19 (2008) 3–17.

[44] Y. Liu, X. Yao, Ensemble learning via negative correlation, Neural Netw. 12 (1999) 1399–1404.

[45] Y. Liu, X. Yao, Simultaneous training of negatively correlated neural networks, IEEE Trans. Syst. Man Cybern. Pt. B-Cybern. 29 (1999) 716–725.

[46] R.A. Jacobs, M.I. Jordan, S.J. Nowlan, G.E. Hinton, Adaptive mixtures of local experts, Neural Comput. 3 (1991) 79–87.

[47] S.E. Yuksel, J.N. Wilson, P.D. Gadar, Twenty years of mixture of experts, IEEE Trans. Neural Netw. Learn. Syst. 23 (2012) 1177–1193.

[48] A. Omari, A .R. Figueiras-Vidal, Feature combiners with gate-generated weights for classification, IEEE Trans. Neural Netw. Learn. Syst. 24 (2013) 158–163.

[49] T. Hastie, R. Tibshirani, Classification by pairwise coupling, Ann. Stat. 26 (1998) 451–471.

[50] K. Duan, S. Keerthi, W. Chu, S. Shevade, A. Poo, Multi-category classification by soft-max combination of binary classifiers, in: Proceedings of the 4th International Conference on Multiple Classifier Systems, Berlin, Heidelberg: Springer-Verlag, 2003, pp. 125–134.

[51] T.G. Dietterich, G. Bakiri, Solving multiclass learning problems via error-correcting output codes, J. Artif. Intell. Res. 2 (1995) 263–286.

[52] D.C. Ciresan, U. Meier, L.M. Gambardella, J. Schmidhuber, Convolutional neural network committees for handwritten character classification, in: Proceedings of the 11th International Conference on Document Analysis and Recognition, New York, NY: IEEE Press, 2011, pp. 1135–1139.

[53] J. Xie, B. Xu, Z. Chung, Horizontal and vertical ensemble with deep representation for classification (2013). arXiv: 1306.2759v1 [cs.LG].

[54] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich, Going deeper with convolutions(2014). arXiv:1409. 4842v1 [cs.CV].

[55] X. Frazão, L.A. Alexandre, Weighted convolutional neural network ensemble, in: Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications, Zug, Switzerland: Springer, 2014, pp. 674–681.

[56] O. Nina, C. Rubiano, M. Shah. Action Recognition using Ensemble of Deep Convolutional Neural Networks. Crcv tech. report, Univ. of Central Florida, 2014.

[57] T. Zhao, Y. Zhao, X. Chen, Building an ensemble of CD-DNN-HMM acoustic model using random forests of phonetic decision trees, in: Proceedings of the IEEE 9th International Symposium on Chinese Spoken Language Processing, 2014, pp. 98–102. Singapore

[58] L. Shao, D. Wu, X. Li, Learning deep and wide: a spectral method for learning deep networks., IEEE Trans. Neural Netw. Learn. Syst. 25 (2014) 2303–2308.
[59] T. Xia, D. Tao, T. Mei, Y. Zhang, Multiview spectral embedding., IEEE Trans. Syst. Man Cybern. Part B 40 (2010) 1438–1446.
[60] A.J. Simpson, Instant learning: parallel deep neural networks and convolutional bootstrapping (2015). arXiv:1505.05972.
[61] S. Lee, S. Purushwalkam, M. Cogswell, D. Crandall, D. Batra, Why M heads are better than one: training a diverse ensemble of deep networks (2015). arXiv:1511.06314v1 [cs.CV].
[62] R.F. Alvear-Sandoval, A.R. Figueiras-Vidal, Does diversity improve deep learn- ing? in: Proceedings of the 23rd European Signal Processing Confer- ence(EUSIPCO), Nice (France), 2015, pp. 2541–2545.
[63] S. Lee, S. Purushwalkan, M. Cogswell, V. Ranjan, D. Crandall, D. Batra, Stochas- tic multiple choice learning for training diverse deep ensembles, Proceedings of the 29th Conference on Advances in Neural Information Processing Systems, Barcelona (Spain), 2016.
[64] N. Srivastava, G.E. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, Dropout: a simple way to prevent neural networks from overfitting, J. Mach. Learn. Res. 15 (2014) 1929–1958.
[65] L. Wan, M. Zeiler, S. Zhang, Y. LeCun, R. Fergus, Regularization of neural net- works using dropconnect, in: Proceedings of the 30th International Conference on Machine Learning, JMLR: W&CP28. Atlanta, GA:, 2013, pp. 1058–1066.
[66] P. Hart, The condensed nearest neighbor rule, IEEE Trans. Inf. Theory 14 (1968) 515–516.
[67] P.W. Munro, Repeat until bored: a pattern selection strategy, in: Proceedings of the 4th Conference on Advances in Neural Information Processing Systems, Morgan Kaufmann. San Mateo, CA:, 1991, pp. 1001–1008.
[68] C. Cachin, Pedagogical pattern selection strategies, Neural Netw. 7 (1994) 171–181.
[69] S.H. Choi, P. Rockett, The training of neural classifiers with condensed datasets, IEEE Trans. Syst. Man Cybern. Pt. B-Cybern. 32 (2002) 202–206.
[70] D. Gorse, A.J. Sheperd, J.G. Taylor, The new ERA in supervised learning, Neural Netw. 10 (1997) 343–352.
[71] A. Lyhyaoui, et al., Sample selection via clustering to construct support vector–like classifiers, IEEE Trans. Neural Netw. 10 (1999) 1474–1481.
[72] S. El-Jelali, A. Lyhyaoui, A.R. Figueiras-Vidal, Designing model based classifiers by emphasizing soft targets, Fundamenta Informaticae 96 (2009) 419–433.
[73] L. Franco, S.A. Cannas, Generalization and selection of examples in feedforward neural networks, Neural Comput. 12 (2000) 2405–2426.
[74] R.F. Alvear-Sandoval, A.R. Figueiras-Vidal, An experiment in pre-emphasizing diversified deep neural classifiers, in: Proceedings of the 24rd European Sym- posium on Artificial Neural Networks, Bruges (Belgium), 2016, pp. 527–532.
[75] R.F. Alvear-Sandoval, M.H. Hayes, A.R. Figueiras-Vidal, Improving denoising stacked auto-encoding classifiers by pre-emphasizing training samples, Neurocomputing (2016). submitted to
[76] Y. LeCun, L. Bouttou, Y. Bengio, P. Haffner, Gradient-based learning applied to document recognition, Proc. IEEE 86 (1998) 2278–2324.
[77] M.A. Nielsen, Neural Networks and Deep Learning, Determination Press (o. l.), 2015.
[78] M. O'Neill, Standard reference data program NIST, 2006, http://www.codeproject.com/kb/library/NeuralNetRecognition.aspx.