Information Fusion 52 (2019) 106–109

Contents lists available at ScienceDirect

Information Fusion

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

On improving CNNs performance: The case of MNIST

1. Introduction

The advancement of Deep Learning Machines (DLMs) along the last decade has permitted to improve the performance of traditional Shallow Learning Machines (SLMs) in the immense majority of application fields. Tutorial contributions [1–3] provide excellent reviews of their history, families, development and applications.

A family of DLMs, Convolutional Neural Networks (CNNs), are very effective to deal with uni- or bi-dimensional signals (such as time registers or images) because their basic architecture consists of applying transversal filters to overlapped portions of the signals and then obtaining the input to the next layer, usually by means of a non-linear step (such as a max-pool), and including a conventional classifier at the top. This architecture is obviously appropriate to perceive the structure of the signal (or image) and, consequently, CNNs are efficient and effective in this kind of classification problems.

SDAE classifiers are an interesting type of representation DLMs: Each layer is initially trained to recover the noise-free version of its input, and its input weights are frozen to train the next layer. In this way, the hidden units are forced to represent more and more useful higher order characteristics of the input. A conventional final classification layer and a refining training complete the design. In our work, we apply a version of the original design which recovers the clean input at each representation step, because it was better when combining it with the other performance improving techniques. It is evident that, contrarily to CNNs, these DLMs do not include any component which can be related to the structure of a signal.

DLMs are one of the avenues that allow to effectively solve the intrinsic limitations of SLMs, that appear in spite of their theoretical unlimited capability to establish input-output transformations [4–6], because the limited information –training examples– which is available for designing them. A previously explored direction to deal with this limitation was to build machine ensembles, using diversity principles to increase the capabilities of SLMs [7–10].

Combination of both kinds of techniques emerged in the first steps of DLMs' expansion period [11,12]. A brief review of the corresponding works is included in [13], where we remark that most of them did not consider classical diversity schemes. In any case, performance results were impressive. For example, Convolutional Neural Network (CNN) ensembles gave misclassification rates of 0.23% [12] with the traditional benchmark MNIST database [14] for handwritten digit recognition, while a drop-connect CNN design offered an even better 0.21% [15]. Our work [13] applied both standard diversity mechanisms (bagging and switching) and binarization of multi-class problems –using the error-correcting code scheme of [16]– to obtain a new record for MNIST, a 0.19% error rate, using also example weighting and data augmentation. Although MNIST is not a challenging problem, we addressed it because we wanted to check if the theoretical advantage of the CNN architecture when approaching the performance limits for a given classification task appeared: Since our basic DLM was a Stacked Denoising Auto-Encoding (SDAE) classifier [14], the answer seemed to be negative. However, at the same time at which we submitted our paper, a new record, 0.17%, was reached with diverse CNN [17]. It must be remarked that, in [17], different final decision architectures and fusion techniques were applied.

An interesting possibility for improving DLMs performance is going deeper, as demonstrated in [18]. Additionally, the recently published work of G. Hinton and his colleagues, Capsule Networks [19–21], postulates that CNN have limited capabilities because their scalar fusion steps and proposes non-scalar representations to increase those capabilities. This is a remarkable point of view: The window weights of a CNN are similar to matched filters and the form of the output is relevant. Experimental results support this perspective. Given the above facts, a question emerges: Is it possible to increase the performance of DLMs by stacking several types of them, expecting that their different characteristics provide some advantage?

In this note, we firstly check if replacing SDAEs in our previous design [13] by CNNs provides some performance advantage in MNIST. At the same time, this serves to evaluate the effectiveness of the representation and disentangling [22] capabilities of SDAEs. After it, we explore if applying an SDAE classifier at the top of the best of the above design further improves the classification performance. If affirmative, it will be not only an additional evidence in favour of the arguments of Sabour et al. [20] and Hinton et al. [21], but also it would suggest that stacking DLMs of different nature, as combining improvement techniques of different character, offers the possibility of obtaining better results.

In Section 2, we will present and discuss extensive experiments to compare the performance of both families –SDAE- and CNN-based– of improved designs when dealing with the MNIST database. In Section 3 we evaluate the results that stacking an SDAE classifier on the best of the previous CNN designs provides. The main conclusions of our work and some suggested lines for further research close the note.

2. First group of experiments

2.1. Experimental framework

MNIST [14] is the database to which we apply the classifiers we propose. Its instances are 28×28, 256-level images of manuscript digits (from 0 to 9), divided into 50,000 train, 10,000 validation, and 10,000 test sets, that also include labels.

We work with a Phyton software[1] which incorporates drop-out and Adam optimizer, as well as with Tensorflow.[2]

We have carried out experiments with several CNNs, but we present only the best performing of them.

[1] https://github.com/tensorflow/tensorflow/tree/r1.4/tensorflow/examples/tutorials/mnist.

[2] https://www.tensorflow.org/.

https://doi.org/10.1016/j.inffus.2018.12.005
Received 27 July 2018; Received in revised form 6 November 2018; Accepted 10 December 2018
Available online 11 December 2018

Table 1
Test error rate, average ± standard deviation, %, for the different CNN-based classifiers. Left column: The same for equivalent SDAE-based classifiers, from [13].

	DLN = SDAE	DLN = CNN
DLN	1.58 ± 0.06	0.39 ± 0.07
PrE+DLN	0.37 ± 0.01	0.32 ± 0.02
DLN+ECOC	–	0.36 ± 0.02
DLN+ECOC + SW	0.36 ± 0.02	0.33 ± 0.01
ECOC+PrE + DLN	–	0.30 ± 0.02
ECOC+PrE + DLN + SW	0.26 ± 0.04	0.18 ± 0.01
ED+ECOC + PrE + DLN	–	0.16 ± 0.01
ED+ECOC + PrE + DLN + SW	0.19 ± 0.01	0.14 ± 0.01

The basic CNN we use is the architecture introduced in [20]. It is as follows: The first layer has 256 branches of 5×5 filters containing a binary valued feature to identify the presence of a digit, and each filter is moved pixel by pixel, providing $24 \times 44 = 576$ output values. The second and third layers have the same structure, their figures being 256 5×5 and $20 \times 20 = 400$ outputs and 128 5×5 filters with $16 \times 16 = 256$ outputs, respectively. After these steps, two fully connected layers of dimensions 328 and 192 are trained to get the desired output, with a 40% drop-out applied to the last layer.

We carry out the same kind of experiments that were done with SDAE classifiers in [13], i.e.:

- A direct application of the single CNN (CNN in Table 1).
- An application of the same single machine but weighting the training examples $\mathbf{x}^{(n)}$ according to

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

  where $o_{ac}^{(n)}$ and $o_{ac'}^{(n)}$ are the outputs of the above classifier (CNN) corresponding to the true class and to the nearest value for the input $\mathbf{x}^{(n)}$, and α, β, $0 \le \alpha, \beta \le 1$, are convex combination parameters that we selected according to the validation set. Form (1) is a general multi-class extension of weighting values we successfully applied for building boosting ensembles [23–25]. This is PrE + CNN in Table 1. Note that this weighting scheme introduces adjustable weights on the samples according to their proximity to the classification border and the error of their preliminary classification by the single CNN, thus allowing to pay more attention to those samples that are more critical for obtaining a good classification performance.
- An ensemble of CNNs applying the ECOC binarization defined by Dietterich and Bakiri code [16] (CNN + ECOC in Table 1).
  ECOC binarization creates a number of binary problems that exceeds the number of classes, where each binary class is composed by some classes of the multi-class problem, in a way that the Hamming distances among the vectors of ideal decisions in high, and therefore there is a margin to correct errors in binary classifications. The implicit redundancy leads to improved performance results with respect to the One-versus-One and One-versus-Rest binarization schemes.
- The same kind of ECOC ensemble as above, but diversifying the full-connected final layer according to the well-know switching method [26]; validating the switching rate and the ensemble size among values {10%, 20%, 30%, 40%} and {11, 21, 51, 101}, respectively. This is CNN + ECOC + SW in Table 1. We apply switching before the last full connected layer.
  Switching creates a number of classifiers by means of randomly changing the labels of a given percentage of each class samples. So, we have a number of different enough problems, but, on the average (or majority), this diversity offers better results than a single machine.
- The ensemble of DLNs resulting from applying the ECOC from the beginning and weighting separately the samples for each dichotomy according to the (binary) form

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

  where $t^{(n)}$ is the target for the sample $\mathbf{x}^{(n)}$, and $o^{(n)}$ the output of the corresponding binary CNN without weighting. We indicate this as ECOC + PrE + CNN in Table 1.
- The denomination ECOC + PrE + CNN + SW in Table 1 corresponds to the addition of switching ensembles at the final layer of each machine of the previous design, with 101 learners each ensemble.

To further improve the performance of the last two designs, a data augmentation process is included. In particular, we apply the Elastic Deformation (ED) method [27], which offered clear advantage in [13]. ED is composed on random horizontal and vertical pixel translations plus a Gaussian smoothing. ED, as other data augmentation techniques, provides additional training examples that can be considered equivalent to the original ones. This increase of examples contributes to a better performance. The corresponding parameters are selected to obtain visually acceptable digits. Then, we have in Table 1:

- ED + ECOC + PrE + CNN, for which the selected augmentation rate is 300%.
- ED + ECOC + PrE + CNN + SW with an augmentation rate of 300%, too.

We apply all the above complementary performance improvement techniques to the basic DLM architectures because all them follow different principles for getting the improvement (sample weighting, redundant binarization, machine diversity, and example augmentation), and, consequently, it can be expected that combining them will offer the possibility of gaining some advantage from each one of these principles. This was true for our application of SDAE classifiers to MNIST [13], but there is an extensive evidence of the advantage of partial combinations in many other papers [17,23–25]. Note that, here, we are applying the same techniques that we applied in [13] to SDAE classifiers, just to check if CNNs maintain their initial advantage with added improvement techniques.

2.2. Results

Table 1 shows the performance results of our experiments. Those corresponding to equivalent designs with SDAE classifiers are included in the left column (they are taken from [13]). Statistics are obtained with 10 runs.

2.3. Discussion

As expected, the single CNN offers better performance than the single SDAE classifier. This advantage decreases when improving modifications (PrE, ECOC, SW, ED) are added, but it remains being relevant. Finally, we arrive to a result, a test error rate of 0.14% on the average for the ED + ECOC + PrE + DLN + SW design, which not only improves the results of the equivalent SDAE structure, but also the previous record, 0.17%, established in [17].

Given this good results, we decided to carry out the second group of experiments: To use an SDAE classifier (which is a machine that carries out a powerful pre-processing of its inputs before trying to classify) to work with the values that the best performing CNN-based architectures, ED + ECOC + PrE + CNN, gives before applying the final layer classifier (switching classifier ensemble). The ensemble diversity will be applied at the output of the SDAE by means of a bagging [28] process, i.e., constructing units of an ensemble by bootstrap resampling of the training set elements.

The operation computational load of the best design is dominated by the number of multiplications at the switching step. There are 15 binary problem branches with 101 elements having $328 + 1$ inputs and 192 nodes, i.e., in the order of 10^7 multiplications. This is a lot compared with the 6×10^4 multiplications that are required by the single

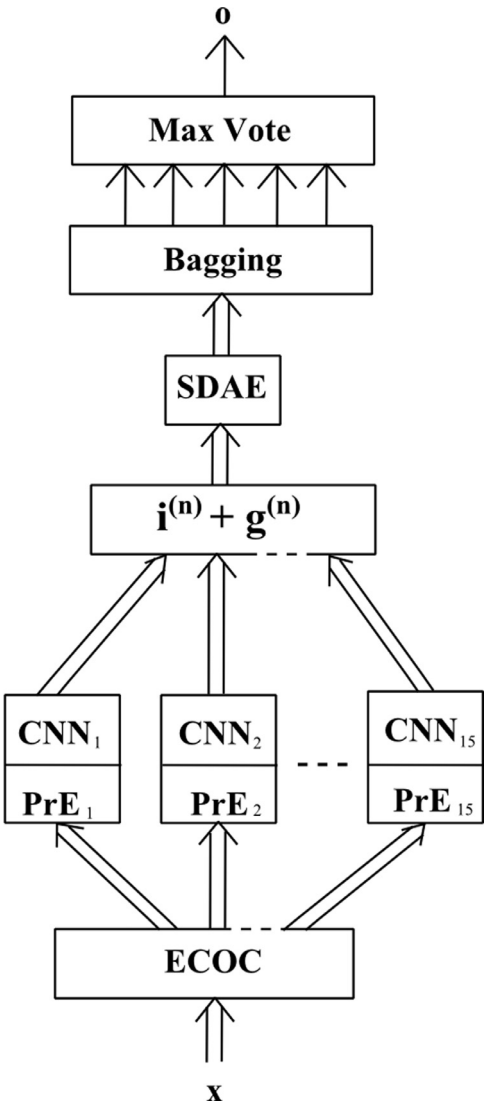

Fig. 1. The overall proposed stacking architecture. The input $\mathbf{x}^{(n)}$ comes from the augmented training set, and the ECOC is the proposed in [16]. Pre-emphasis forms for the dichotomies and CNNs are as described in the text, as well as the SDAE architecture and training (input $\mathbf{i}^{(n)}$ plus noise $\mathbf{g}^{(n)}$). 10-class bagging provides the input to a majority vote among its maximum outputs: This vote is the final output $\mathbf{o}^{(n)}$.

CNN, but the improvement is very important. The comparison with the $15\times101\times10^6$ multiplications that correspond to the equivalent SDAE design is also very favorable.

3. Experiments with the stacked architecture

3.1. Experimental framework

We delete the SW classification layers of a trained ED + ECOC + PrE + CNN + SW structure and the last layer units, and we inject the resulting 15×328 CNN output values into the first layer of our SDAE, which has 960 units. Three more SDAE layers with 1000 units lead to the final 10-classes classification step. Each new layer is added to obtain at its output the free-of-noise input. A zero-mean, Gaussian random value is added to each input vector component for the noisy training, its variance, 10% of that of the component, being determined by the validation set results.

SDAE's last layer output is applied to an ensemble of one hidden-layer MLP bagging classifiers [28] with 1000 hidden units and a ten component soft-max output. The final decision is taken by the majority of the highest outputs of all these MLPs. The bootstrap population size, B, and the number of MLPs in the bagging ensemble, M, are selected among {60%, 80%, 100%, 120%} and {21, 51, 101, 121}. As expected, there are saturation effects. No fine tuning is done.

Fig. 1 presents the final architecture we use.

3.2. Results

The best performance result is obtained for $B = 100\%$ and $M = 121$: The test error rate is $0.13 \pm 0.00\%$ (50 runs average). This permits to say that this value is a new record in MNIST classification.

3.3. Discussion

It must be accepted that to replace the simple MLP switching final decision by the more powerful SDAE plus bagging structure slightly improves the MNIST classification performance. This also indicates that stacking DLMs of different nature can be fruitful, as deeper architectures are in many applications. Futhermore, these results reveal that CNN architectures are not completely effective, because sophisticate posterior decision machines provide performance improvements.

With respect to the computational load of this design, it can be approximated by the number of multiplications in the bagging units, which is 121×10^6, i.e., around 10^8. We pay an order of magnitude of computational load to reach the classification record.

4. Conclusions and further work

In this note, we have experimentally checked that to employ appropriate CNN basic blocks in machine ensembles that also use other mechanisms (pre-emphasis, binarization, diversity, and elastic distortion) to improve performance when classifying MNIST provides advantage with respect to employing SDAE-based equivalent schemes.

Futhermore, when an SDAE with a bagging ensemble of final classifiers is stacked over the (best) CNN scheme, replacing its final decision layer, a slight but clear improvement appears, that leads to these designs to reach an absolute record. This fact permits to ensure that not in all cases DLMs extract all the information to solve a processing task, and that stacking different kinds of DLMs can be fruitful to get better results.

To check if "capsule" DLMs can be successfully diversified and improved by means of the complementary techniques we used here and to explore what DLM stacking combinations are useful for different kinds of data problems are immediate research directions that we are exploring at the present time.

Note

The software blocks we have developed for our experiments and links to other software blocks we have used are available at http://www.tsc.uc3m.es/~ralvear/Software.htm.

Acknowledgements

This work has been partly supported by research grants CASI-CAM-CM (S2013/ICE-2845, Madrid Community/FEDER, EUSF) and Macro-ADOBE (TEC2015-67719, MINECO/FEDER, EU). The authors thanks the anonymous reviewers for their useful comments and suggestions.

Supplementary material

Supplementary material associated with this article can be found, in the online version, at doi:10.1016/j.inffus.2018.12.005.

References

[1] Y. Bengio, Learning deep architectures for ai, Found. Trends Mach. Learn. 2 (2009) 1–127.
[2] J. Schmidhuber, Deep Learning in Neural Networks: An Overview, Technical Report IDSIA-03-14, University of Lugano, 2014. arXiv: 1404.7828v4 [cs.NE].
[3] L. Deng, D. Yu, Deep learning: methods and applications, Found. Trends Mach. Learn. 7 (2014) 197–387.
[4] G. Cybenko, Approximation by superpositions of a sigmoidal function, Math. Control Signals Syst. 2 (1989) 303–314.
[5] K. Hornik, M. Stinchcombe, H. White, Multilayer feedforward networks are universal approximators, Neural Netw. 2 (1989) 359–366.
[6] V. Vapnik, Statistical Learning Theory, Wiley, New York, NY, 1998.
[7] A.J. Sharkey, Combining Artificial Neural Nets: Ensemble and Modular Multi-Net Systems, Springer, London, UK, 1999.
[8] L.I. Kuncheva, Combining Pattern Classifiers: Methods and Algorithms, Wiley, Hoboken, NJ, 2004.
[9] L. Rokach, Pattern Classification Using Ensemble Methods, World Scientific, Singapore, 2010.
[10] R.E. Schapire, Y. Freund, Boosting: Foundations and Algorithms, MIT Press, Cambridge, MA, 2012.
[11] D.C. Cireşan, U. Meier, L.M. Gambardella, J. Schmidhuber, Convolutional neural network committees for handwritten character classification, in: Proc. 11th International Conference on Document Analysis and Recognition, IEEE Press, New York, NY, 2011, pp. 1135–1139.
[12] D. Cireşan, U. Meier, J. Schmidhuber, Multi-column deep neural networks for image classification, in: Proc. Conference on Computer Vision and Pattern Recognition, IEEE Press, New York, NY, 2012, pp. 3642–3649.
[13] R.F. Alvear-Sandoval, A.R. Figueiras-Vidal, On building ensembles of stacked denoising auto-encoding classifiers and their further improvement, Inf. Fusion 39 (2018) 41–52.
[14] 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.
[15] L. Wan, M. Zeiler, S. Zhang, Y. LeCun, R. Fergus, Regularization of neural networks using dropconnect, in: Proc. 30th International Conference on Machine Learning, JMLR Proc., Atlanta, GA, 2013, pp. 1058–1066.
[16] T.G. Dietterich, G. Bakiri, Solving multiclass learning problems via error-correcting output codes, J. Artif. Intell. Res. 2 (1995) 263–286.
[17] A. Loquercio, F. Della Torre, M. Buscema, Computational eco-system for handwritten digits recognition, arXiv:1703.01872v1 [stat.ML] (2017).
[18] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich, Going deeper with convolutions, arXiv:1409.4842v1 [cs.CV] (2015).
[19] G.E. Hinton, A. Krizhevsky, S.D. Wang, Transforming auto-encoders, in: International Conference on Artificial Neural Networks, Springer, 2011, pp. 44–51.
[20] S. Sabour, N. Frosst, G.E. Hinton, Dynamic routing between capsules, in: Advances in Neural Information Processing Systems, 2017, pp. 3856–3866.
[21] G.E. Hinton, S. Sabour, N. Frosst, Matrix capsules with em routing, in: Proc. International Conference on Learning Representations, 2018, pp. 2859–3869. Vancouver, Canada
[22] P.P. Brahma, D. Wu, Y. She, Why deep learning works: a manifold disentanglement perspective., IEEE Trans. Neural Netw. Learn. Syst. 27 (10) (2016) 1997–2008.
[23] 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.
[24] 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.
[25] A. Ahachad, L. Álvarez-Pérez, A.R. Figueiras-Vidal, Boosting ensembles with controlled emphasis intensity, Pattern Recognit. Lett. 88 (2017) 1–5.
[26] L. Breiman, Randomizing outputs to increase prediction accuracy, Mach. Learn. 40 (2000) 229–242.
[27] M. ONeill, Standard reference data program NIST, http://www.codeproject.com/kb/library/NeuralNetRecognition.aspx (2006).
[28] L. Breiman, Bagging predictors, Mach. Learn. 24 (1996) 123–140.

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

José L. Sancho-Gómez
GTTS-TDAM/DTIC, Universidad Politécnica de Cartagena, Spain

Aníbal R. Figueiras-Vidal
GAMMA-L+/DTSC, Universidad Carlos III de Madrid, Spain

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

Received 27 July 2018
Revised 6 November 2018
Accepted 10 December 2018