DOES DIVERSITY IMPROVE DEEP LEARNING?

R. F. Alvear-Sandoval, A. R. Figueiras-Vidal

GAMMA-L+/DTSC, Universidad Carlos III de Madrid
{r.f.alvear.s@gmail.com, arfv@tsc.uc3m.es}

ABSTRACT

In this work, we carry out a first exploration of the possibility of increasing the performance of Deep Neural Networks (DNNs) by applying diversity techniques to them. Since DNNs are usually very strong, weakening them can be important for this purpose. This paper includes experimental evidence of the effectiveness of binarizing multi-class problems to make beneficial the application of bagging to Denoising Auto-Encoding-Based DNNs for solving the classical MNIST problem.

Many research opportunities appear following the diversification idea: We mention some of the most relevant lines at the end of this contribution.

Friendly dedicated to Prof. Giovanni Sicuranza, with deep admiration and sincere appreciation.

Index Terms— Auto-encoding, classification, depth, diversity.

1. INTRODUCTION

1.1. Preliminaries

Digital Signal Processing (DSP) has provided practical solutions to a lot of different problems in many fields –from geophysics to health, including mechanics, environment, computation, communications, and biology, to mention just a few. Maybe signal transmission has been one of the areas which contributed in a relevant mode to the development of DSP. Since most physical communication channels are esentially linear and time-invariant, frequency played a key role. Linear filtering was the research focus. Adaptive (linear) filters emerged mainly to deal with lack of knowledge, and subsequently to cope with non-stationarity. Equalization and echo cancellation serve as well-known examples. [1] was the starting point for all this.

Yet even in transmission some cases appeared for which non-linearity was unavoidable, such as satellite communications. And other related tasks –not only coding/decoding, but speech and object recognition– are not satisfactorily solvable by linear approaches. These facts provoked an increasing interest in non-linear filtering. But, as wisely Mathews and Sicuranza said in [2], a milestone along this research avenue, two obstacles required sustained efforts: Implementation complexity and understanding difficulties. We, human beings, are reluctant to work with ideas that our mind does not interpret easily –curiously enough, our perception is clearly non-linear.

On the other hand, early Learning Machines (LMs), especially Neural Networks (NNs), became usable [3]. Contrarily to one of the most popular families of non-linear architectures, the Linear In the Parameter (LIP) filters, that consists of linear combinations of fixed non-linear transformations, NNs are based on trainable transformations, such as sigmoidal versions of linear combinations –the well-known Multi-Layer Perceptrons (MLPs). The difficulties with the LIP filters is how to select the fixed transformations, and sizing and training for NNs. Things become worse when real-time adaptation is needed: It is easy for a given LIP filter, but to select its transformations to work in an unpredictable non-stationary environment is a serious challenge. Real-time adaptation is intrinsecally difficult for NNs, and only recently some ideas on how to do it can be considered as preliminary steps for this objective [4].

1.2. Diversity and depth

Until few years ago, the immense majority of MLP designs were shallow, i.e., with one or –exceptionally– two hidden layers. The one-hidden-layer architectures being theorically universal approximators [5, 6], available training examples are finite and controlled approximations to optimal designs are impossible. To close the wall, although the basic training Back-Propagation (BP) algorithm and its modifications and extensions permit theoretically to train deep architectures (DNNs), the difficulties in weight initialization and the many minima that cost landscapes show occluded this solution.

Fortunately, to select convenient (repetitive) architectures, such as Convolutional Neural Networks (CNNs) [7], or to apply auxiliary techniques to gain depth before training to solve the addressed problem, as well as some algorithmic recent advances, make the design of DNNs affordable [8–10]. And the

This work has been partially supported by Community of Madrid research project grant S2013/ICE-2845 CASI-CAM-CM

978-0-9928626-3-3/15/$31.00 ©2015 IEEE

performances that these LMs offer are impressive: See some records in [9].

A different way to gain expressive power in LM designs is to build ensembles. They consist of a number of (relatively simple) diverse learners whose outputs are aggregated, as firstly proposed in [11]. Training is carried out separately in some cases, or jointly in others, such as in boosting [12], whose fame is well deserved. Let us emphasize that learners' diversity is the key for success. There is not room here even for a brief presentation of the main diversification techniques, but [13, 14] are pretty complete textbooks on the subject.

In our humble opinion, to conceive, to design, to evaluate and to apply highly expressive –and adaptive, when needed– LMs is today's main challenge for signal and data processing: Many unsolved relevant problems are non-linear and difficult –and some of them, non-stationary. And when thinking about how to increment the expressive power of LMs, a natural question must be answered: ***Does diversity improve deep learning?*** I.e., can we get advantage if we combine diversity and depth?

1.3. Diversity plus depth

There have been some previous efforts to answer the above question. Multi-Column CNNs (MC-CNNs) [15, 16] applied image subsizing and different predistortions to induce diversity. In [17], trainable linear/softmax combinations were used to aggregate the outputs of MC ensembles. Sequentially different problems are created in Deep Stacking (Convex) Nets (DSNs) [18, 19] by injecting the available outputs as inputs for each new (deeper) learner. In [20], consecutive video frames serve to create optical flows to design deep CNN ensembles. Using multiple triphone states to train context-dependent DNN ensembles for speech recognition is studied in [21]. And the multiview spectral embedding technique of [22] is employed in [23] for diversifying the aggregation of some DNNs.

The mentioned diversification attempts being sucessful, and MC and DSN designs gaining more and more attention, they do not constitute a systematic approach to the use of accredited standard ensemble design methods with DNNs: All the proposed diversification methods are "ad hoc", and not attempt of applying standard techniques has been carried out. In this article, we present some preliminary results of our work in that direction. We remark that our present objective is not to beat performance records, but exploring possibilities for it in a future. We wish to understand how to efficiently combine diversity and depth by analyzing and discussing some experimental results.

The rest of this contribution is structured as follows. In Section 2 we introduce the problem to be addressed in the experiments, as well as the basic diversification method and DNNs to be applied. The experiments and their results are sequentially presented in Section 3, in order to permit an ordered discussion of the difficulties we face and potential solutions for them. The conclusions of this preliminary work, some insights for further research, and selected avenues to be explored close the paper.

2. DATABASE, BASIC DIVERSITY, AND DNNS FOR THE EXPERIMENTS

2.1. MNIST

MNIST [7] is a classical database for DNNs' benchmarking experiments. It represents handwritten digits by 784 (28x28), 256 (grey) levels variables. It consists of 50000 training, 10000 validation and 10000 test (labeled) samples. The best classification result for this problem is an error rate 0.2%, approximately, a slightly superhuman performance.

2.2. Bagging

Bagging ("**B**ootstrap and **agg**regat**ing**") [24] separately trains a number N of unstable learners, each one with a different bootstrap sample of the training set. Their outputs are aggregated by means of simple non-trainable schemes, such as their decisions' majority vote. We remark that unstability means that diverse training sets produce parameter values that are different enough to generate outputs whose aggregation reduces their implicit "training noises", improving the overall performance.

2.3. Denoising Auto-Encoding Based Deep Neural Networks

Denoising Auto-Encoding Based Deep Neural Networks (D-AEB-DNNs) [24] are deep MLPs that are constructed layer-by-layer in the following manner. The first layer is expansive –its size is greater than the size of the input– and serves to provide an output whose target is the corresponding input sample. To avoid the identity operation, some noise is added to the inputs, keeping the output targets in their original form. After BP training, the output layer is removed, and the process is repeated with MLPs of the same hidden layer size until a certain depth. Then a classification layer with softmax activations for multi-class problems indicates the result. A final refining supervised training is usually applied.

3. EXPERIMENTS AND THEIR DISCUSSION

3.1. Experimental framework

We used the software available in [25], mentioned in [26]. The DNN architecture was the same that Vincent et al. employed in [27] (called SDAE-3 there): 3x1000 auto-encoding layer plus the final 10-softmax classification. Training partly followed Vincent et al.'s experimental practice: Gradient steps 0.01 for the first layer and 0.02 for the rest and for

refining, and 40 epochs (that are enough for convergence). However, we were not able of reaching the performance in [27] using 25% added noise, and our best results correspond to 10% noise level: % error rate average $\pm$ standard deviation, 1.58 ± 0.06 (figures in [27] are 1.28 ± 0.22).

With respect to bagging, the non-trainable parameter values we explored were N$\in\{25, 50, 100\}$ and bootstrap sizes B$\in\{60,80,100,120\}$ (% training set size). To alleviate the high computational cost of these designs, we first designed 150 bagged versions for each value of B, and we built the ensembles by randomly selecting N of them. 10 random selections are carried out to average the results. Obiously, this reduces the effective diversification, but, we repeat, our objective is just to determine if diversity improves DNNs performance.

3.2. Direct application of bagging

The first experiments we carried out tried to directly apply bagging to the selected D-AEB-DNN architecture. They failed. Even the "omniscient" trick –selecting the values of non-trainable parameters N and B according to the test set, which is an improper design procedure, but useful to appreciate the potential limits of the method being applied –gave an error rate 1.63 ± 0.01 (N=100, B=120: Clear saturation effects appeared for these parameters). Oviously, no improvement is obtained.

A close look into these experiments indicated that the obstacle can be the following: An error rate 1.58% is very good for a 10-class problem, showing that the used D-AEB-DNN elements are very strong, i.e., they perform so well that, even being unstable, there is not room for taking advantage from diversification, because their outputs still tend to be too similar. So, weakening the learners seemed to offer an opportunity.

3.3. The One vs. One approach

To reduce the expressive power of the D-AEB-DNN learners –reducing their size– constitutes an evident option for weakening them. But, in this first exploration, we preferred to avoid the required high computational effort for it. Consequently, we considered a simpler alternative: To binarize the 10-class problem –which is also a diversification technique.

Binarization is the conversion of a multi-class problem into a number of binary problems whose results indicate the multi-class solution. Its most elementary forms are One vs. Rest (OvR), in which C classifiers are trained to separate the samples of each one of C classes from the other samples, and One vs. One (OvO) [28], that designs the C(C-1)/2 binary classifiers solving each possible dichotomy. Since 10(10-1)/2=45 designs are affordable, we selected OvO for our preliminary experiments. It can be said that OvO is superior to OvR in general terms, specially when OvR faces imbalanced problems (very different sizes of dichotomy populations), that is the case even for balanced datasets when C is relatively high. The information reduction for training serves to reduce the strongness of the corresponding D-AEB-DNNs.

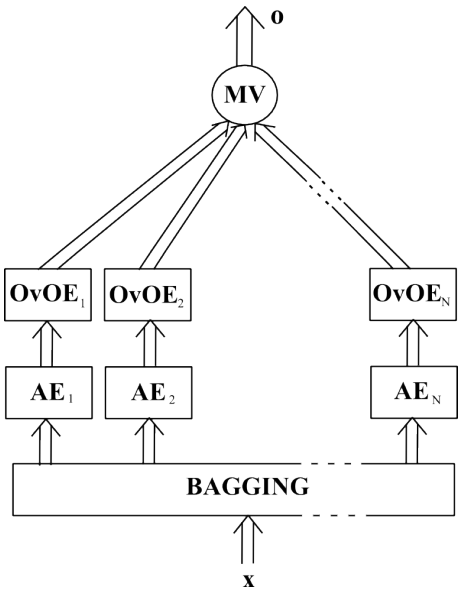

Fig. 1: D-AEB-OvO-DNN architecture for multi-class problems. AE_n are the diverse (deep) auto-encoding steps, and $OvOE_n$ the One vs. One ensembles. MV means majority vote, $\mathbf{x}$ is the input, and $\mathbf{o}$ is a class indicator.

Before presenting our experimental results, we must emphasize that Dietterich and Bakiri [29] perceived that multi-class binarization is equivalent to design error correcting codes for transmission, hence the name Error Correcting Output Codes (ECOCs) that the subsequent ensembles receive. There is not space here even for a brief overview of ECOCs, but [12] dedicates a chapter to this subject. In any case, it is clear that using adequate ECOCs surely will allow to improve the performance of the designs we are proposing.

Figure 1 presents the AEB-DNNs' ensemble we use for this second group of experiments, in which bagging is again applied, but final classifiers are OvO implementations.

Table 1 shows the performance results we obtained using OvO at the classification level of the D-AEB-DNNs, designs we call D-AEB-OvO-DNNs.

First of all, let us say that, as expected, the OvO monolithic design provides a small, but clear, improvement with respect to its multi-class counterpart. Secondly: Bagging diversity becomes effective for OvO designs. Validation selects N=100, B=100, and N=100, B=120, and the corresponding test performances are 0.85 ± 0.01 and 0.86 ± 0.01, almost a 50% reduction with respect to the original D-AEB-DNN and more than 30% lower than the test performance of the monolithic D-AEB-OvO-DNN. Let us insist on the fact that OvO simply produces a moderate improvement, but it allows bag-

a

N \ B	60	80	100	120
25	2.50±0.04	1.73±0.20	1.65±0.09	1.65±0.08
50	1.65±0.10	1.07±0.08	1.02±0.10	1.00±0.10
100	1.27±0.05	0.82±0.04	**0.70±0.02**	**0.70±0.01**

b

D-AEB-OvO-DNN	1.40±0.06			
N \ B	60	80	100	120
25	2.61±0.08	1.81±0.21	1.72±0.12	1.72±0.12
50	1.73±0.12	1.11±0.12	1.04±0.13	1.02±0.12
100	1.31±0.01	0.86±0.00	***0.85±0.01***	***0.86±0.01***

Table 1: Performance results (% error rate average ± standard deviation) for the D-AEB-OvO-DNN machine and bagging ensembles using it. a) Validation set, b) test set. Best results are in boldface, validated design results are in italics

ging to be effective, further decreasing the error rate.

The saturation effects that appear when N or B increase are really important: They appear in the same regions of values of these non-trainable parameters both for validation and test, and, therefore, they do not only indicate that there is not need of exploring bigger values, but also this parallel flatness means that validation will select good values for the final design. Note that the selected design performance and the "omniscient" result do not show any relevant difference.

4. CONCLUSIONS AND FURTHER WORK

In this paper, we have shown that to weaken DNNs is a possibility to gain performance advantage by combining them with standard diversification techniques, such as bagging. In particular, binarization of multi-class problems constitutes an attractive improvement opportunity. Of course, much more experimental work is needed to determine what DNN designs and diversification procedures are really useful for different kinds of problems: Do not forget that strongness can be an obstacle, and that it depends on the dataset under study. We can add that preliminary results of other experiments we are carrying out (using switching [30] as diversification technique, diversifying only at the final classification step, etc.) are also offering advantageous results.

Among all the new research directions that this work opens, we consider that to explore

- the effects of approppiate ECOC forms
- architectural weakening of DNNs, and its potential combination with ECOC
- other diversification methods
- the possibility of diversifying only the final (classification) layer of DNNs
- if other improvement mechanisms can be advantageously combined with these diversification ideas

is fundamental to allow designing really "Big Learning" machines, one of the tools (adaptation is the other) which is required to face some difficult and relevant Big Data problems: The "Big Problems" that justify this kind of effort. We cordially offer our collaboration for this purpose.

REFERENCES

[1] B. Widrow and M.E. Hoff, "Adaptive switching circuits", in *IRE WESCON Conv. Rec.*, vol. 4, pp. 96–104, New York, NY, 1960.

[2] V.J. Mathews and G.L. Sicuranza, *Polynomial Signal Processing*, New York, NY: Wiley, 2000.

[3] D.E. Rumelhart, J.L. McClelland, and the PDP Research Group, "Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Vol. 1: Foundations", Cambridge, MA: MIT Pres, 1986.

[4] Special issue on learning in nonstationary and evolving environments, *IEEE Trans. Neural Networks and Learning Sys*, vol. 25, no. 1, Jan. 2014.

[5] G. Cybenko, "Approximation by superpositions of a sigmoidal function", *Mathematics of Control, Signals and Sys.*, vol. 2, pp. 303–314, 1989.

[6] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators", *Neural Networks*, vol. 2, pp. 359–366, 1989.

[7] Y. LeCun et. al., "Backpropagation applied to handwritten zip code recognition", *Neural Computation*, vol. 1, pp. 541–551, 1989.

[8] Y. Bengio, "Learning deep architectures for AI", *Foundations and Trends in Machine Learning*, vol. 2, pp. 1–127, 2009.

[9] J. Schmidhuber, "Deep learning in neural networks: An overview", T. R. IDSIA-03-04, Univ. of Lugano, 2014; arXiv:1404.7828v4 [cs.NE].

[10] L. Deng and D. Yu, *Deep Learning: Methods and Applications*, Hanover, MA: Now Publishers, 2013.

[11] L.-K. Hansen and P. Salamon, "Neural network ensembles", *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 12, pp. 993–1001, 1990.

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

[13] L. Rokach, *Pattern Classification Using Ensemble Methods*, Singapore: World Scientific, 2009.

[14] C. Zhang and Y. Ma, *Ensemble Machine Learning:*

Methods and Applications, New York, NY: Springer, 2012.

[15] D.C. Ciresan, U. Meier, L.M. Gambardella, and J. Schmidhuber, "Convolutional neural network committees for handwritten character classification", in *Proc. 11th Intl. Conf. Document Analysis and Recognition*, pp. 1135–1139, New York, NY: IEEE Press, 2011.

[16] D.C. Ciresan, U. Meier, and J. Schmidhuber, "Multi-column deep neural networks for image classification", in *Proc. Conf. Computer Vision and Pattern Recognition*, pp. 3642–3649, New York, NY: IEEE Press, 2012.

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

[18] L. Deng and D. Yu, "Deep convex net: A scalable architecture for speech pattern classification", in *Proc. Interspeech 2011*, pp. 2285–2288. Florence, Italy, 2011.

[19] B. Hutchinson, L. Deng, and D. Yu, "A deep architecture with bilinear modeling of hidden representations: Applications to phonetic recognition", in *IEEE Intl. Conf. Acoustics, Speech and Signal Proc.*, pp. 4805–4808, New York, NY, 2012.

[20] O. Nina, C. Rubiano, and M. Shah, "Action recognition using ensemble of deep convolutional neural networks", in memkite.com/deep-learning-bibliography, 2014.

[21] T. Zhao, Y. Zhao, and X. Chen, "Building an ensemble of CD-DNN-HMM acoustic model using random forests of phonetic decision trees", in *9th Intl. Symposium on Chinese Spoken Language Proc.*, pp. 98–102, Singapore, 2014.

[22] T. Xia, D. Tao, T. Mei, and Y. Zhang, "Multiview spectral embedding", *IEEE Trans. Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 40, pp. 1438–1446, 2010.

[23] L. Shao, D. Wu, and X. Li, "Learning deep and wide: A spectral method for learning deep networks", *IEEE Trans. Neural Networks and Learning Sys.*, vol. 25, pp. 2303–2308, 2014.

[24] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, "Extracting and composing robust features with denoising autoencoders", in *Proc. 25th Intl. Conf. Machine Learning*, pp. 1096–1103, New York, NY: ACM Press, 2008.

[25] www.cs.toronto.edu/~hinton/MatlabForSciencePaper.html

[26] G.E. Hinton and R.R. Salakhutdinov, "Reducing the dimensionality of data with neural networks", *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[27] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol, "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion", *J. Machine Learning Res.*, vol. 11, pp. 3371–3408, 2010.

[28] T. Hastie and R. Tibshirani, "Classification by pairwise coupling", *The Annals of Statistics*, vol. 26, pp. 451–471, 1998.

[29] T.G. Dietterich and G. Bakiri, "Solving multiclass learning problems via error-correcting output codes", *J. Artificial Intelligence Res.*, vol. 2, pp. 263–286, 1995.

[30] L.Breiman, "Randomizing outputs to increase prediction accuracy", *Machine Learning*, vol. 40, pp. 229–242, 2000.