ESANN 2015 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 22-24 April 2015, i6doc.com publ., ISBN 978-287587014-8.
Available from http://www.i6doc.com/en/.

Resource-efficient incremental learning in very high dimensions

Alexander Gepperth[1], Mathieu Lefort[1], Thomas Hecht[1] and Ursula Körner[2] *

1- ENSTA ParisTech/UIIS lab and INRIA FLOWERS
828 Boulevard des Maréchaux, 91762 Palaiseau, France and
200 avenue de la Vieille Tour, 33405 Talence Cedex

2- Honda Research Institute Europe GmbH
Carl-Legien-Str.30, 73076 Offenbach am Main, Germany

Abstract. We propose a three-layer neural architecture for incremental multi-class learning that remains resource-efficient even when the number of input dimensions is very high (≥ 1000). This so-called projection-prediction (PROPRE) architecture is strongly inspired by biological information processing in that it uses a prototype-based, topologically organized hidden layers trained with the SOM learning rule controlled by a global, task-related error signal. Furthermore, the SOM learning adapts only the weights of localized neural sub-populations that are similar to the input, which explicitly avoids the catastrophic forgetting effect of MLPs in case new input statistics are presented to the architecture. As the readout layer uses simple linear regression, the approach essentially applies locally linear models to "receptive fields" (RF) defined by SOM prototypes, whereas RF shape is implicitly defined by adjacent prototypes (which avoids the storage of covariance matrices that gets prohibitive for high input dimensionality). Both RF centers and shapes are jointly adapted w.r.t. input statistics and the classification task. Tests on the MNIST dataset show that the algorithm achieves compares favorably compared to the state-of-the-art LWPR algorithm at vastly decreased resource requirements.

1 Introduction

Incremental learning remains a challenging issue in machine learning. While it is almost self-evident to biologists that learning should be incremental, the technical realization presents baffling difficulties. First of all, incremental learning is inherently sub-optimal when it comes to optimizing an objective (or loss) function. As one can never assume to have seen all training samples at any single point during training, optimization can only take into account the examples seen up to the present moment. Furthermore, the statistics of input-output relations are usually not homogeneous for any finite dataset, so incremental learning must essentially assume non-stationary input statistics at some time scale, which raises the question of how to fuse already learned aspects of a task, without destroying them, with new ones. The latter issue is a real problem for connectionist models

*Thomas Hecht gratefully acknowledges financial support by the French Armaments Procurement Agency (DGA) and Ecole Polytechnique.

ESANN 2015 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 22-24 April 2015, i6doc.com publ., ISBN 978-287587014-8.
Available from http://www.i6doc.com/en/.

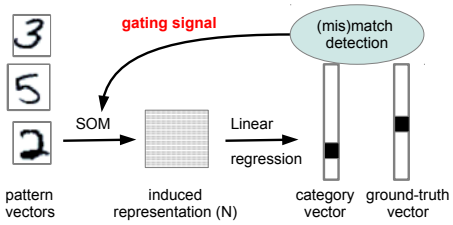

Fig. 1: Schema of the three-layer PROPRE architecture composed of input, induced and output representations. Initially there is a forward transmission step, propagating the information to the top-level of the hierarchy where it is decided whether a correct result was obtained. In case it was not, SOM weights are updated in the feedback step, thus leading to a representation of difficult samples in N.

of learning [1] and has been termed "catastrophic forgetting", and it is clear that any feasible incremental learning algorithm needs to avoid this issue.

Biological foundations As biological incremental learning has reached a high degree of perfection, we explicitly investigated the biological literature for hints as how to this might be achieved. Basing ourselves on observations from the basic sensory cortices, we noted that sensory representations seem to be prototype-based, where prototype-sensitive neurons are topologically arranged by similarity [2, 3, 4, 5]. Learning seems to act on these representations in a task-specific way, where more prototypes are allocated to sensory regions where finer discrimination is necessary [6], a mechanism which presumably gated learning through acetylcholine release in case of task failures [7, 8]. In particular, learning seems to respect and even generate topological layout of prototypes by changing only a small subset of neural selectivities [9], namely those neurons who previously best matched the stimuli to be learned [5].

Model properties We propose a three-layer neural model for incremental learning that contains a topologically organized representation of prototypes in its hidden layer (termed "induced representation"), trained by the self-organized map (SOM) algorithm [10]. Due to the properties of SOMs, learning is always strictly local in the sense that only the prototypes that are similar to the best-matching one are adapted, thus avoiding catastrophic forgetting. SOM learning is activated by adverse task performance, which conversely means that learning stops once the task is acquired, thus maintaining long-term stability. Classification is performed by simple linear regression from the hidden layer towards a population-coded target vector after first applying a non-linear transfer function to all hidden layer activities.

Related work Incremental learning algorithms are especially interesting for robotics applications [11], and in fact several very interesting proposals have already been

ESANN 2015 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 22-24 April 2015, i6doc.com publ., ISBN 978-287587014-8.
Available from http://www.i6doc.com/en/.

made in this context [11]. An especially popular algorithm in robotics is LWPR [12], which partitions the input spaces into receptive fields (RFs), volumes that are defined by a centroid and a covariance matrix, to which separate linear models are applied. Many other incremental algorithms, reviewed in [11] partition the input space in a similar way and thus will presumably run into memory problems when input dimensionality is high, as LWPR does.

Contribution of this article In this article, we will propose a model for incremental learning that can cope with scenarios where $KM \sim 10000$ (K, M denoting input.output dimensionality) and beyond, and evaluate its performance on the well-known MNIST benchmark [13]. We perform a comparison to LWPR that explicitly evaluates the incremental aspect of learning by training the algorithms on a subset of classes and subsequently adding the remaining classes.

2 Methods

2.1 The PROPRE architecture

PROPRE is an architecture composed of different algorithmic modules, rather than an algorithm in itself. One PROPRE iteration consists of the following steps, as described in [14], where only the computation of the predictability measure λ is changed to represent the current binary classification error:
input: new data is fed into the input representation I and provided to the SOM, and a new target representation T is provided **projection:** activity is formed in the induced representation N (see Fig. 1) by projection of I onto the SOM prototypes **prediction:** based on activity in N, a linear regression step is performed to produce representation P which predicts class membership **evaluation:** a mismatch measure is computed between P and T **update:** linear regression weights are updated. SOM weights are updated only if mismatch was detected. In mathematical terms, the whole model is governed by the following equations, where we denote neural activity at position $\vec{y} = (a, b)$ in a 2D representation X by $z^X(\vec{y}, t)$ and weight matrices for SOM and LR, represented by their line vectors attached to target position $y = (a, b)$ by $w_{\vec{y}}^{\mathrm{SOM}}$:

$$z^N(\vec{y}, t) = w_{\vec{y}}^{\mathrm{SOM}}(t) \cdot z^I(t) \tag{1}$$

$$z^P(\vec{y}, t) = w_{\vec{y}}^{\mathrm{LR}}(t) \cdot z^I(t) \tag{2}$$

$$\lambda(t) = 0 \text{ if } \mathrm{argmax}_{\vec{y}} z^P(\vec{y}, t) = \mathrm{argmax}_{\vec{y}} z^T(\vec{y}, t), 1 \text{ else} \tag{3}$$

$$w_{\vec{y}}^{\mathrm{LR}}(t+1) = w_{\vec{y}}^{\mathrm{LR}}(t) + 2\epsilon^{\mathrm{LR}} z^I(t) \left(z^P(t) - z^T(t)\right) \tag{4}$$

$$w_{\vec{y}}^{\mathrm{SOM}}(t+1) = \text{ norm} \left(w_{\vec{y}}^{\mathrm{SOM}}(t) + \lambda(t)\epsilon^{\mathrm{SOM}} g_\sigma(\vec{y} - \vec{y}^*)(z^I - w_{\vec{y}}^{\mathrm{SOM}})\right) \tag{5}$$

$$\tag{6}$$

where $g_\sigma(\vec{x})$ is a zero-mean Gaussian function with standard deviation σ and $\vec{y}^*$ denotes the position of the best-matching unit (the one with the highest activity) in N. In accordance with standard SOM training practices, the SOM learning

ESANN 2015 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 22-24 April 2015, i6doc.com publ., ISBN 978-287587014-8.
Available from http://www.i6doc.com/en/.

rate and radius, ϵ^{SOM} and σ, start at ϵ_0, σ_0 and are exponentially decreased in order to attain their long-term values $\epsilon_\infty, \sigma_\infty$ at $t = T_{\text{conv}}$.

2.2 LWPR

We use the LWPR algorithm as described in [12] using a publicly available implementation[15].

2.3 The MNIST handwritten digit database

For all experiments, we use he publicly available MNIST classification benchmark as described in [13]. It contains 10 classes, corresponding to the 10 handwritten digits from "0" to "9", see also Fig. 1. Each sample has a dimensionality of $K = 28 \times 28 = 784$. We split the data into two sets: D_{0-4} containing the digits from "0" to "4", and D_{5-9} containing the remaining digits. Each set is again split, at a proportion of 5:1, into a training and a test set to measure generalization performance, giving in total four data sets: D^{train}_{0-4} (25.000 samples), D^{test}_{0-4} (5.000 samples), and analogously D^{train}_{5-9}, D^{test}_{5-9}. For training and evaluating performance on all digits, we also create the sets D^{train}_{0-9}, D^{test}_{0-9} in an analogous fashion.

3 Experiments

Experimental setup We conduct an identical set of experiments both for PROPRE and for LWPR, which is designed to measure the capability to perform incremental learning. To this effect, we present D^{train}_{0-4} for $T_1 = 15000$ iterations, subsequently D^{train}_{5-9} for $T_2 = 15000$ iterations, and lastly D^{train}_{0-9} for $T_3 = 2000$ iterations, which is fair to both algorithms as it allows them to converge (longer training times did not change results). At all times, we can measure generalization performance on any of the sets D^{test}_{0-9}, D^{test}_{0-4} and D^{test}_{5-9}. In order to establish a baseline performance, to be compared to offline, batch type algorithms, we also train and evaluate both algorithms on D_{0-9}. We use the following parameters for PROPRE: $n = 30$, $\epsilon^{\text{SOM}} = 0.2$, $\epsilon^{\text{LR}} = \frac{0.09}{n^2}$, $\epsilon_0 = 0.8$, $\sigma_0 = 0.6n$, $T_1 = 4000$, $\epsilon_\infty = 0.2$ and $\sigma_\infty = 1$. Both SOM and LR weight matrices were initialized to random uniform values between -0.001 and 0.001. For PROPRE, input vectors where always normalized to have an L2 norm of 1.0 before presenting them. LWPR is parametrized as follows, using notation from [15] for details: init_alpha=0, init_D =3.2 (in order to limit receptive field number to ¡ 8), diag_only=0, update_D=1, using default parameters everywhere else. No normalization of inputs was performed for LWPR.

Results Results are given in Tab. 1 and show that baseline performance of both algorithms is roughly comparable, with a slight edge to PROPRE. Both algorithms convincingly avoid catastrophic forgetting since performance on D^{test}_{0-9} after successive training on D^{train}_{0-4} and D^{train}_{5-9} is $>> 10\%$ which we would expect in case the digits 0-4 had been forgotten. In both cases performance improves

ESANN 2015 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 22-24 April 2015, i6doc.com publ., ISBN 978-287587014-8. Available from http://www.i6doc.com/en/.

	PROPRE			LWPR		
test set / train set	D_{0-9}^{test}	D_{0-4}^{test}	D_{5-9}^{test}	D_{0-9}^{test}	D_{0-4}^{test}	D_{5-9}^{test}
D_{0-9}^{train}	90%	x	x	83%	x	x
D_{0-4}^{train}	43%	95%	x	44%	93%	x
D_{5-9}^{train}	80	x	x	71	x	x
D_{0-9}^{train}	85	x	x	80	x	x%

Table 1: Performance evaluation of PROPRE and LWPR on MNIST data. First row: performance on D_{0-9}^{test} when both algorithms are trained on all classes simultaneously, e.g., on D_{0-9}^{train}. Rows 3,4,5: performance when successively training on D_{0-4}^{train} (row 3), D_{5-9}^{train} (row 4) and, for a small interval on D_{0-9}^{train} (row 5).

when performing a short retraining on D_{0-9}^{train}. For PROPRE, this is because linear regression models for D_{0-4}^{train} are no longer fully valid after training on D_{5-9}^{train}, and retraining can help fix this. Similarly, for LWPR, some classes share receptive fields and therefore the associated linear models are no longer fully valid after training on D_{0-4}^{train} and need to be readjusted.

4 Discussion

Generally, LWPR had to be parametrized very carefully in order not to exceed the computer memory limits, allowing at most 8 receptive fields per output dimension. This is because, for K input dimensions, a single receptive field in LWPR requires roughly $5K^2$ floating point values, mainly in order to store the covariance matrix and keep track of data statistics. As LWPR allocates RFs independently for each of M output dimensions, the overall memory requirements are $\mathcal{O}(5MK^2)$ which gets problematic for large K as it is the case for MNIST($K = 28^2$). Limiting RF creation surely limited LWPR's ability to exert its full potential on this benchmark; on the other hand, it is a fair comparison because both algorithms were executed on the same computer (Ubuntu Linux, 2 GB RAM, CoreI7 processor) and had to make do with the same resources. PROPRE suffered from no memory limitations since, for a fixed size $n \times n$ of the induced representation N, PROPRE requires $Kn^2 + n^2M = n^2(K + M)$ floating point values for storing the weight matrices. If we compare the baseline performance of both algorithms to the performances of other algorithms on MNIST, we find that PROPRE performs better than some but significantly worse than the best algorithms, whereas LWPR (with these specific parameters) performs worse than even linear models. For the case of PROPRE, this is the price to pay for online and incremental learning capacity, as already amply discussed in [12]. For LWPR, it can be concluded that in the present form it is unsuited for this kind of input/output dimensionalities. The fact that LWPR is intended to perform regression and not classification does not influence memory requirements, and in addition classification is a special case of regression even if the reverse does not hold.

ESANN 2015 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 22-24 April 2015, i6doc.com publ., ISBN 978-287587014-8. Available from http://www.i6doc.com/en/.

5 Conclusion

We have presented an algorithm for resource-efficient incremental learning that draws its efficiency from principles of biological information processing. We showed that it compares favorably with the quasi-standard algorithm for incremental learning, LWPR, when tested on a standard machine learning benchmark while requiring only a fraction of computational, and above all, memory resources. What is more, the bulk of computation is contained in the SOM projection implemented by a matrix multiplication, which is an operation that can is easily parallelized. Therefore, not only memory consumption hut also runtime speed can be very low when using PROPRE. Next steps will include tests on robotic applications (e.g., online learning of forward and inverse models) as well as a hierarchical versions the architecture with multiple stacked induced representations N for improving computational power.

References

[1] Ian J Goodfellow, Mehdi Mirza, Xia Da, Aaron Courville, and Yoshua Bengio. An empirical investigation of catastrophic forgeting in gradient-based neural networks. *arXiv preprint arXiv:1312.6211*, 2013.

[2] Keiji Tanaka. Inferotemporal cortex and object vision. *Annual review of neuroscience*, 19(1):109–139, 1996.

[3] David A Leopold, Igor V Bondar, and Martin A Giese. Norm-based face encoding by single neurons in the monkey inferotemporal cortex. *Nature*, 442(7102):572–575, 2006.

[4] David A Ross, Mickael Deroche, and Thomas J Palmeri. Not just the norm: Exemplar-based models also predict face aftereffects. *Psychonomic bulletin & review*, 21(1):47–70, 2014.

[5] Cynthia A Erickson, Bharathi Jagadeesh, and Robert Desimone. Clustering of perirhinal neurons with similar properties following visual experience in adult monkeys. *Nature neuroscience*, 3(11):1143–1148, 2000.

[6] Daniel B Polley, Elizabeth E Steinberg, and Michael M Merzenich. Perceptual learning directs auditory cortical map reorganization through top-down influences. *The journal of neuroscience*, 26(18):4970–4982, 2006.

[7] Norman M Weinberger. The nucleus basalis and memory codes: Auditory cortical plasticity and the induction of specific, associative behavioral memory. *Neurobiology of Learning and Memory*, 80(3):268 – 284, 2003. Acetylcholine: Cognitive and Brain Functions.

[8] Michael E Hasselmo. The role of acetylcholine in learning and memory. *Current opinion in neurobiology*, 16(6):710–715, 2006.

[9] Edmund T Rolls, GC Baylis, ME Hasselmo, and V Nalwa. The effect of learning on the face selective responses of neurons in the cortex in the superior temporal sulcus of the monkey. *Experimental Brain Research*, 76(1):153–164, 1989.

[10] T Kohonen. Self-organized formation of topologically correct feature maps. *Biol. Cybernet.*, 43:59–69, 1982.

[11] Olivier Sigaud, Camille Salaün, and Vincent Padois. On-line regression algorithms for learning mechanical models of robots: a survey. *Robotics and Autonomous Systems*, 59(12):1115–1129, 2011.

[12] Sethu Vijayakumar, Aaron D'souza, and Stefan Schaal. Incremental online learning in high dimensions. *Neural computation*, 17(12):2602–2634, 2005.

[13] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. In *Intelligent Signal Processing*, pages 306–351. IEEE Press, 2001.

[14] A Gepperth. Efficient online bootstrapping of representations. *Neural Networks*, 2012.

[15] Sethu Vijayakumar Stefan Klanke and Stefan Schaal. A library for locally weighted projection regression. *Journal of Machine Learning Research (JMLR)*, 9:623–626, 2008.