Learning Deep Belief Networks from Non-Stationary Streams

Roberto Calandra[1], Tapani Raiko[2], Marc Peter Deisenroth[1], and Federico Montesino Pouzols[3]

[1] Fachbereich Informatik, Technische Universität Darmstadt, Germany
[2] Department of Information and Computer Science, Aalto University, Finland
[3] Department of Biosciences, University of Helsinki, Finland

Abstract. Deep learning has proven to be beneficial for complex tasks such as classifying images. However, this approach has been mostly applied to static datasets. The analysis of non-stationary (e.g., concept drift) streams of data involves specific issues connected with the temporal and changing nature of the data. In this paper, we propose a proof-of-concept method, called Adaptive Deep Belief Networks, of how deep learning can be generalized to learn online from changing streams of data. We do so by exploiting the generative properties of the model to incrementally re-train the Deep Belief Network whenever new data are collected. This approach eliminates the need to store past observations and, therefore, requires only constant memory consumption. Hence, our approach can be valuable for life-long learning from non-stationary data streams.

Keywords: Incremental Learning, Adaptive Learning, Non-stationary Learning, Concept drift, Deep Learning, Deep Belief Networks, Generative model, Generating samples, Adaptive Deep Belief Networks.

1 Introduction

Machine learning typically assumes that the underlying process generating the data is stationary. Moreover the dataset must be sufficiently rich to represent this process. These assumptions do not hold for non-stationary environments such as time-variant streams of data (e.g., video). In different communities, a number of approaches exist to deal with non-stationary streams of data: Adaptive Learning [4], Evolving Systems [2], Concept Drift [15], Dataset Shift [12]. In all these paradigms, incomplete knowledge of the environment is sufficient during the training phase, since learning continues during run time.

Within the adaptive learning framework, there is a new set of issues to be addressed when dealing with large amounts of continuous data online: limitations on computational time and memory. In fast changing environments, even a partially correct classification can be valuable.

Since its introduction [8, 3] Deep Learning has proven to be an effective method to improve the accuracy of Multi-Layer Perceptrons (MLPs) [6]. In particular, Deep Belief Networks (DBNs) have been well-established and can be considered the state of the art for artificial neural networks. To the best of our knowledge, DBNs have not been used to incrementally learn from non-stationary streams of data. Dealing with changing streams of data with classical DBNs requires to store at least a subset of the previous observations, similar to other non-linear approaches to incremental learning [10, 1]. However, storing large amounts of data can be impractical.

The contributions of this paper are two-fold. Firstly, we study the generative capabilities of DBNs as a way to extract and transfer learned beliefs to other DBNs. Secondly, based on the possibility to transfer knowledge between DBNs, we introduce a novel approach called Adaptive Deep Belief Networks (ADBN) to cope with changing streams of data while requiring only constant memory consumption. With the ADBN it is possible to use the DBN parameters to generate observations that mimic the original data, thus reducing the memory requirement to storing only the model parameters. Moreover, the data compression properties of DBNs already provide an automatic selector of the relevant extracted beliefs. To the best of our knowledge, the proposed ADBN is the first approach toward generalizing Deep Learning to incremental learning.

2 Deep Belief Networks

Deep Belief Networks are probabilistic models that are usually trained in an unsupervised, greedy manner. DBNs have proven to be powerful and flexible models [14]. Moreover, their capability of dealing with high-dimensional inputs makes them ideal for tasks with an innate number of dimensions such as image classification. The basic building block of a DBN is the Restricted Boltzmann Machine (RBM) that defines a joint distribution over the inputs and binary latent variables using undirected edges between them. A DBN is created by repeatedly training RBMs stacked one on top of the previous one, such that the latent variables of the previous RBM are used as data for the next one. The resulting DBN includes both generative connections for modeling the inputs, and recognition connections for classification (see [8] for details).

One possible use of DBNs is to initialize a classical MLP: While a DBN works with activation probabilities of binary units, they can be re-interpreted as continuous-valued signals in an equivalent MLP. This "pre-training" proves to be better than random initialization and has been shown to lead to consistent improvements in the final classification accuracy [7, 6]. A second use for DBNs is dimensionality reduction. In this case, a classifier can be trained "on top" of the DBN, which means that the input of the classifier is nothing else but the data in the reduced space (i.e., the DBN output). This second configuration (from now on referred as DBN/classifier) has the advantage that the generative capabilities of the DBN are maintained.

DBNs typically require the presence of a static dataset. To deal with changing streams, the usage of DBNs would require storing all the previous observations to re-train the DBN. For infinite-lasting streams of data this would translate into an infinite memory requirement and increasing computational time for training.

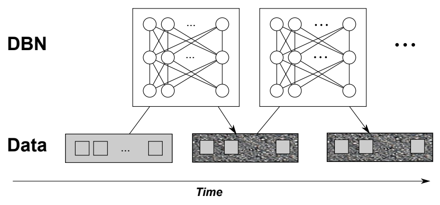

Fig. 1: Regenerative Chaining: Alternately learning a DBN from data and generating data from a DBN.

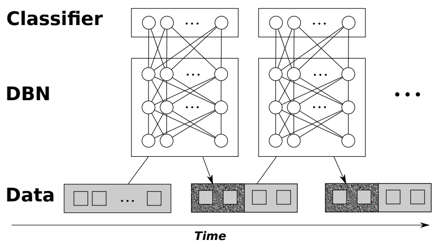

Fig. 2: ADBN: DBN+classifier are trained from both the generated samples and newly incoming data.

3 Adaptive Deep Belief Networks

To address the limitations of DBNs for non-stationary streams of data (memory consumption, training time), we propose a novel approach based on the generative capabilities of a DBN. Three novel contributions are presented in this section in a logical sequence where each of them extend upon the previous. At first, we investigate the possibility of using samples generated from a DBN to transfer the learned beliefs (i.e., knowledge) to a second DBN. Then we show how to extend this approach to transfer not only unsupervised but also supervised knowledge (i.e., including labels). Finally, we present our novel approach called Adaptive Deep Belief Networks (ADBN). This approach is based on transferring supervised knowledge by means of generated samples and, jointly, learns new knowledge from the novel observations.

Belief Regeneration An interesting feature of DBNs is the capability of generating samples. These samples can be considered a representation of the beliefs learned during the training phase [8]. Under this assumption we can exploit the generated samples as an approximation of the knowledge of the DBN. We propose to train a second *regenerated* DBN from the samples generated from a trained *original* DBN. From this procedure, which we call *Belief Regeneration*, we theoretically obtain an equivalent model to the original DBN.

This procedure can be iterated by training an n^{th} DBN from the samples generated by the $(n-1)^{th}$ regenerated DBN as shown in Fig. 1. We call this procedure *Regenerative Chaining*, and a DBN generated from n repetitions of Belief Regeneration is an nth-generation DBN.

Classifier Regeneration DBNs can generate unlabeled samples that mimic the distribution of the training inputs. When making use of the DBN/classifier configuration it is still possible to generate unlabeled samples with the generative connections. Furthermore, these samples can be used as a standard input for the recognition connections and, thus, be classified. Hence, this procedure allows the generation of datasets of labeled samples. Similarly to Belief Regeneration, we use this artificially generated dataset to train a second DBN/classifier,in what we call *Classifier Regeneration.*Chaining the Classifier Regeneration process is the building block for ADBNs.

Adaptive Deep Belief Networks When dealing with non-stationary streams of data, there is a need to consider two different aspects. While it is necessary to retain past knowledge, new one must also be incorporated as well. We saw how generated labeled samples can be used to repeatedly reconstruct both the DBN and the classifier that approximate the original ones. The DBN/classifier regeneration can effectively keep acquired knowledge in our model even when discarding the past observations (i.e., the dataset). We propose to exploit this belief transfer to generalize DBNs to an incremental learning paradigm. In order to incorporate also new knowledge in the model, we can use both generated samples and novel data from the stream, for the re-train of the DBN/classifier, as shown in Fig. 2. The use of such training data allows the DBN/classifier to incorporate new knowledge while retaining old one. Moreover, the memory consumption is constant as after each training period all the previous data (both artificially generated and real) are discarded.

4 Experiments

To evaluate the properties of our models we used the hand-written digit recognition MNIST dataset [11] in our experiments. The MNIST dataset consists of a training set of 60000 observations and a test set of 10000 observations where every observation is an image of 28x28 binary pixels stored as a vector (no spatial information was used).

To train the RBMs, we used the algorithm introduced by Cho et al. [5] that makes use of Contrastive Divergence learning (CD-1). We used Gibbs sampling to generate samples from a trained DBN. The reconstruction error over a dataset is defined as

$$R(\mathbf{X}) = \tfrac{1}{N} \sum\nolimits_{i=1}^{N} \sum\nolimits_{j=1}^{D} (X_{ij} - \hat{X}_{ij})^2 \,, \tag{1}$$

where N is the number of observations, D the dimensionality of the input X, and $\hat{X}$ is the reconstructed input. For fine-tuning the neural network, we used the Resilient Propagation (Rprop) algorithm [13]; and in particular the IRprop variant [9][1]. We used the Log-sigmoid as transfer function and the Mean Squared Error as error function to train the network.

Belief Regeneration To experimentally evaluate the Belief Regeneration process, a DBN with topology [784-600-500-400] was trained. Fig. 3 shows how the number of samples used to train the regenerated DBN influences the reconstruction error in Eq. (1). A higher number of samples better approximates the original DBN trained with the full dataset. However, it is also computationally expensive to generate many samples. In our experiment, there seems to be a clear threshold at 750 samples above which the original DBN can be considered well approximated. A further indication is given by the visual inspection of the generated

[1] Our implementation is available at `http://www.mathworks.com/matlabcentral/fileexchange/32445-rprop`

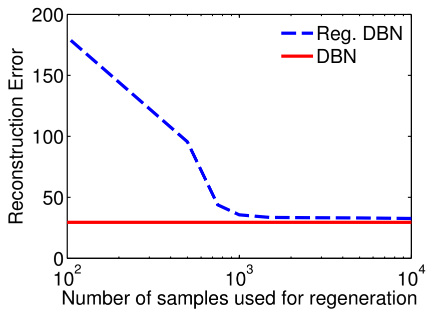

Fig. 3: Reconstruction error of the MNIST test set using a DBN regenerated with a varying amounts of generated samples.

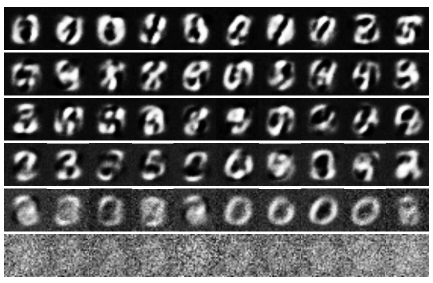

Fig. 4: MNIST samples generated from DBNs: first row from original DBN, then from DBN regenerated with 10000, 2500, 750, 500, and 100 generated samples, respectively.

samples from the regenerated DBN in Fig. 4. Above 750 samples there is little difference between the samples generated from original and reconstructed DBN (top row of Fig. 4), for a human observer. Fig. 5 shows that for chained regenerations the reconstruction error gradually increases with the number of sequential reconstructions. Similar conclusions are visually drawn from the generated samples in Fig. 6 where after 100 generations of regeneration (using 10000 samples at each generation) there is a visible degradation in the quality of the generated samples. The reason of this degradation is the error propagation between sequential regenerations.

However, fine-tuning a 100th generation DBN shows little decrease in terms of classification accuracy compared to fine-tuning the original DBN, as shown in Fig. 7. This result suggests that despite becoming humanly incomprehensible (Fig. 6), the generated samples retain valuable features for training a DBN and still prove to be useful during an eventual fine-tuning: Fine-tuning initialized from a regenerated DBN led to a similar optimum as the original DBN.

Classifier Regeneration Using a DBN/classifier allows us to generate labeled samples. Examples of the generated samples and respective labels are shown in Fig. 8. These artificially generated datasets are used to train subsequent DBNs/ classifiers, as described in Sec. 3. Fig. 9 shows the classification accuracies of the regenerated DBNs/classifiers after n generations. While the decrease in performance is consistent, we are using samples *generated* from our model. Furthermore, the number of samples is only a fraction of the original dataset size.

Adaptive Deep Belief Networks We trained a DBN and classifier using 3 digits (8,6,4) of the MNIST dataset. Every 50 fine-tuning iterations, we presented a new batch of data containing samples from a novel digit to the ADBN. These samples, together with the generated ones, were then used to re-train both the DBN and the classifier, see Sec. 3. Fig. 10 shows the classification accuracy and memory consumption of the ADBN on all 10 digits when adding new digits to the

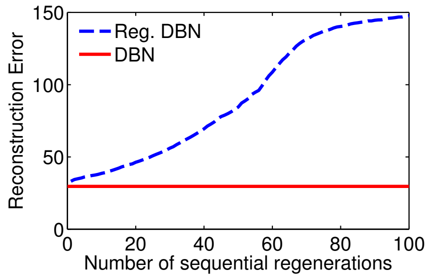

Fig. 5: Reconstruction error of the MNIST test set using a DBN regenerated a varying amounts of times. Despite the increase in reconstruction error, the classification accuracies of fine-tuning do not change (see Fig. 7).

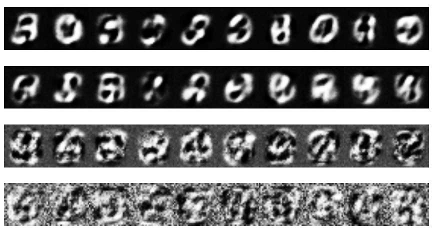

Fig. 6: MNIST samples generated from DBNs: first row from the original DBN, then for 1, 50 and 100th-generation DBNs. Despite the degeneration, even after 100 generations the samples retain useful features to train a new DBN.

data set. The accuracy increases, which means that the ADBN can successfully learn from new data while at the same time retaining the knowledge previously acquired. In Fig. 10, we compare to a DBN that is trained on all the previous observations which led to a higher classification accuracy but at the expense of the memory consumption. While the DBN memory increase linearly (as we store more and more observations), the amount of memory required by the ADBN is constant: Only the model parameters need to be stored.

5 Conclusions and Discussion

In this paper, we presented the Adaptive Deep Belief Networks, a promising approach that generalizes DBNs to deal with changing streams of data (e.g., number of classes and shift in distribution) while requiring only constant memory.

ADBNs can be incrementally trained by means of Belief Regeneration. Belief Regeneration consists of iteratively transferring beliefs between DBNs by training a second DBN using samples from the *original* DBN.

In ADBNs, new data can be integrated into the already acquired beliefs retraining the DBN with both the novel and artificially generated samples. While the classification accuracy suffers compared to fully trained DBNs, the ADBN does not need to store past observations. Hence, the memory requirements are constant since only the model parameters have to be stored. Moreover, unlike other Adaptive Learning methods, the ADBN is a generative model and can deal with high-dimensional data.

In our approach the generative capabilities and the sampling method adopted are of great importance. Although the Contrastive divergence (CD) learning (used in this work), produces good classifiers, it does not work as well as a

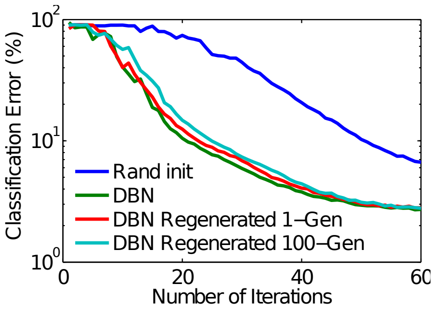

Fig. 7: Classification accuracies on the MNIST test set during the fine-tuning. Using regenerated DBNs does not substantially decrease the classification accuracy compared to the original DBN.

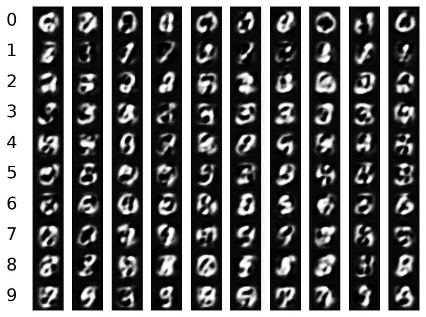

Fig. 8: MNIST samples generated from DBNs, labeled and then used for regeneration: every row corresponds to labeled samples.

generative model. We therefore believe that the use of more appropriate training and sampling scheme could be beneficial.

Many adaptive learning approaches make use of explicit mechanisms to forget the past. To this regard, ADBNs present no explicit mechanism to forget selected observations. Instead, the less representative observations are naturally forgotten during the regeneration process. The choice of the number of samples to use for each epoch of the ADBN training can be a sensitive parameter. In particular, the ratio between generated samples and novel observations, can be used to modify the stability/plasticity trade-off.

Finally, an interesting extension to our approach is the possibility to change the topology of the network adaptively at running time in order to adapt the capability to the complexity of the environment.

Acknowledgments. We thank Olli Simula and Jan Peters for invaluable discussions and the friendly environments they provided to realize this paper. The research leading to these results has received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement #270327 and by the DFG within grant #SE1042/1.

References

1. C. C. Aggarwal, J. Han, J. Wang, and P. S. Yu. On demand classification of data streams. In *Proceedings of KDD 2004*, pages 503–508.
2. P. Angelov, D. P. Filev, and N. Kasabov. *Evolving Intelligent Systems: Methodology and Applications.* Wiley-IEEE Press, 2010.
3. Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle. Greedy layer-wise training of deep networks. In *Proceedings of NIPS 2006*, volume 19, pages 153–160.
4. A. Bifet, editor. *Proceeding of the 2010 Conference on Adaptive Stream Mining: Pattern Learning and Mining from Evolving Data Streams.*

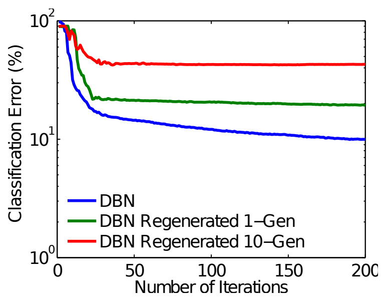

Fig. 9: Comparison of the classification accuracies on the MNIST test set during the training of the classifier on top, using the original and regenerated classifiers after different N-generations.

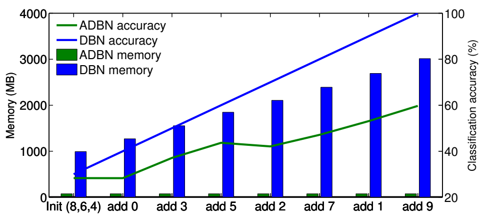

Fig. 10: Classification accuracy and memory consumption of the DBN and the ADBN on all 10 digits during the training phase: Initially trained with 3 digits (8,6,4), for every novel digit introduced the DBN/classifier is regenerated. A classical DBN achieves higher classification accuracy but at the expense of memory consumption. In contrast, the ADBN requires only a constant memory.

5. K. Cho, T. Raiko, and A. Ilin. Enhanced gradient and adaptive learning rate for training restricted Boltzmann machines. In *Proceedings of ICML 2011*, pages 105–112.
6. D. Erhan, A. Courville, Y. Bengio, and P. Vincent. Why does unsupervised pre-training help deep learning? In *Proceedings of AISTATS 2010*, pages 201–208.
7. D. Erhan, P.-A. Manzagol, Y. Bengio, S. Bengio, and P. Vincent. The difficulty of training deep architectures and the effect of unsupervised pre-training. In *Proceedings of AISTATS 2009*, pages 153–160.
8. G. E. Hinton, S. Osindero, and Y. W. Teh. A fast learning algorithm for deep belief nets. *Neural Computation*, 18(7):1527–1554, 2006.
9. C. Igel and M. Hüsken. Improving the RPROP learning algorithm. In *Proceedings of NC 2000*, pages 115–121.
10. M. Last. Online classification of nonstationary data streams. *Intelligent Data Analysis*, 6(2):129–147, 2002.
11. Y. LeCun and C. Cortes. MNIST handwritten digit database. `http://yann.lecun.com/exdb/mnist/`, 2010.
12. J. Quiñonero Candela, M. Sugiyama, A. Schwaighofer, and N. D. Lawrence. *Dataset Shift in Machine Learning*. The MIT Press, 2009.
13. M. Riedmiller and H. Braun. A direct adaptive method for faster backpropagation learning: the RPROP algorithm. In *IEEE International Conference on Neural Networks*, pages 586–591 vol.1, 1993.
14. R. Salakhutdinov. *Learning deep generative models*. PhD thesis, University of Toronto, 2009.
15. I. Zliobaite. Learning under concept drift: an overview. *CoRR*, 2010.