Why does Unsupervised Pre-training Help Deep Discriminant Learning?

Dumitru Erhan, Yoshua Bengio, Aaron Courville
Pierre-Antoine Manzagol, Pascal Vincent
University of Montreal

Abstract

Recent research has been devoted to learning algorithms for deep architectures such as Deep Belief Networks and stacks of auto-encoder variants, with impressive results obtained in several areas. The best results obtained on supervised learning tasks involve an unsupervised learning component, usually in an unsupervised pre-training phase, with a generative model. Even though these new algorithms have enabled training deep models fine-tuned with a discriminant criterion, many questions remain as to the nature of this difficult learning problem. The main question investigated here is the following: why does unsupervised pre-training work and why does it work so well? Answering these questions is important if learning in deep architectures is to be further improved. We propose several explanatory hypotheses and test them through extensive simulations. We empirically show the influence of unsupervised pre-training with respect to architecture depth, model capacity, and number of training examples. The experiments confirm and clarify the advantage of unsupervised pre-training. The results suggest that unsupervised pre-training guides the learning towards basins of attraction of minima that are better in terms of the underlying data distribution; the evidence from these results supports an unusual regularization explanation for the effect of pre-training.

1 Introduction

Deep learning methods aim at learning feature hierarchies with features from higher levels of the hierarchy formed by the composition of lower level features. They include learning methods for a wide array of *deep architectures*, including neural networks with many hidden layers (Vincent et al., 2008) and graphical models with many levels of hidden variables (Hinton et al., 2006), among others (Zhu et al., 2009; Weston et al., 2008). Theoretical results, reviewed and discussed by Bengio and LeCun (2007), suggest that in order to learn the kind of complicated functions that can represent high-level abstractions (e.g. in vision, language, and other AI-level tasks), one may need *deep architectures*. The recent surge in experimental work in the field seems to support this notion, accumulating evidence that in challenging AI-related tasks – such as computer, vision, natural language processing (NLP), robotics, or information retrieval – deep learning methods significantly out-perform comparable but shallow competitors, and often match or beat the state-of-the-art.

These recent demonstrations of the potential of deep learning algorithms were achieved despite the serious challenge of training models with many layers of adaptive parameters. In virtually all instances of deep learning, the objective function is a highly non-convex function of the parameters, with the potential for many distinct *local minima* in the model parameter space. The principal difficulty is that not all of these minima provide equivalent generalization errors and, we suggest, that for deep architectures, the standard training schemes tend to place the parameters in regions of the parameters space that generalize poorly – as was frequently observed empirically but rarely reported (Bengio & LeCun, 2007).

The breakthrough to effective training strategies for deep architectures came in 2006 with the algorithms for training deep belief networks (DBN) (Hinton et al., 2006) and stacked auto-encoders (Ranzato et al., 2007; Bengio et al., 2007), which are all based on a similar approach: greedy layer-wise unsupervised pre-training followed by supervised fine-tuning. Each layer is pre-trained with an unsupervised learning algorithm, learning a nonlinear transformation of its input (the output of the previous layer) that captures the main factors of variation in its input. This unsupervised pre-training sets the stage for a final training phase where the deep architecture is fine-tuned with respect to a discriminant training criterion with a gradient-based optimization. While the improvement in performance of trained deep models offered by the pre-training strategy is impressive, little is understood about the mechanisms underlying this success.

The objective of this work is to explore, through extensive experimentation, how unsupervised pre-training works to render learning deep architectures more effective and why they appear to work so much better than traditional neural network training methods. There are a few reasonable hypotheses why pre-training might work. One possibility that unsupervised pre-training acts as a kind of regularizer, putting the parameter values in the appropriate range for discriminant training. Another possibility, suggested by Bengio et al. (2007), is that pre-training initializes the model to a point in parameter space that somehow renders the optimization process more effective, in the sense of achieving a lower minimum of the empirical cost function.

Here, we argue that our experiments support a view of pre-training as an unusual form of *regularization*: minimizing variance and introducing bias towards configurations of the parameter space that correspond to useful transformations of the input (i.e. extracting features and factors) from the point of view of modeling the input distribution. The corresponding prior is that some of these factors are also useful for performing a target classification. This perspective places pre-training well within the family of recently developed semi-supervised methods. The pre-training approach is, however, unique among semi-supervised training strategies in that it acts by defining a particular initialization point for standard supervised training rather than either modifying the supervised objective function (Barron, 1991) or explicitly imposing constraints on the parameters throughout training (Lasserre et al., 2006). This type of initialization-as-regularization strategy has precedence in the neural networks literature, in the shape of the early stopping idea (Sjöberg & Ljung, 1995; Amari et al., 1997). We suggest that, in the highly non-convex situation of training a deep architecture, defining a particular initialization point *implicitly* imposes constraints on the parameters in that it specifies which minimum (of a very large number of possible minima) of the cost function is sought. In this way, it may be possible to think of the pre-training as being related to the approach of Lasserre et al. (2006).

Another important and distinct property of the pre-training strategy is that in the standard situation of training using stochastic gradient descent, the beneficial generalization effects due to pre-training do not appear to diminish as the number of labeled examples grows very large. We argue that this is a consequence of the combination of the non-convexity (multi-modality) of the objective function and the dependency of the stochastic gradient descent method on example ordering. We found that early changes in the parameters (when they are still small) have a greater impact on the final region (basin of attraction of the descent procedure) in which the learner ends up. In particular, unsupervised pre-training sets the parameter in a region from which better basins of attraction can be reached, both from the standpoint of the training criterion (especially for large datasets) and in terms of generalization. Hence, although pre-training is a regularizer, it can have a positive effect on the training objective when the number of training examples is large.

2 The Challenges of Deep Learning

We start with a perspective on why standard training of deep models through gradient backpropagation appears to be so difficult. First, it is important to establish what we mean in stating that training is difficult. What complicates matters is that different levels in the hierarchy may be more or less difficult to train. For example the training criterion is generally convex in the parameters of the top layer, and training one-hidden layer neural networks is generally very successful (in spite of the potential for local minima), hence the higher layers (near the output) are easy to optimize, given the lower layers. However, given the higher layers as fixed, we conjecture that training the lower layers is a difficult optimization problem, and trying to optimize both simultaneously may be even more difficult. Furthermore, because with enough capacity the top two layers can easily overfit the training set, training error does not necessarily reveal the difficulty in optimization the lower layers. This is where unsupervised pre-training seems to make a big difference, by setting parameters of the lower layers in a region from which good solutions (in terms of generalization error) can be found.

We believe the central challenge in training deep architectures is dealing with the strong dependencies that exist during training between the parameters across layers. One way to conceive the difficulty of the problem is that we must simultaneously:

1. adapt the lower layers in order to provide adequate input to the final (end of training) setting of the upper layers
2. adapt the upper layers to make good use of the final (end of training) setting of the lower layers.

The second problem is easy on its own (i.e., when the final setting of the other layers is known). It is not clear how difficult is the first one, and we conjecture that a particular difficulty arises when both sets of layers must be learned jointly, as the gradient of the objective function is limited to a local measure given the current setting of other parameters. As shown in our experiments here, the standard training schemes tend to place the parameters in regions of the parameters space that generalize poorly.

A separate but related issue appears if we focus our consideration of traditional training methods for deep architectures on stochastic gradient descent. A sequence of examples along with an online gradient descent procedure

defines a trajectory in parameter space, which converges in some sense (the error does not improve anymore, maybe because we are near a local minimum). The hypothesis is that small perturbations of that trajectory (either by initialization or by changes in which examples are seen when) have more effect early on. Early in the process of following the stochastic gradient, changes in the weights tend to increase their magnitude and the amount of non-linearity of the network. As this happens, the set of regions accessible by stochastic gradient descent on samples of the training distribution becomes smaller. This is consistent with the idea that there are basins of attraction of the dynamics of learning, and that early on small perturbations allow to switch from one basin to a nearby one, whereas later on (typically with larger parameter values), it is unlikely to "escape" from such a basin of attraction. Hence the early examples can have a larger influence and, in practice, trap the model parameters in particular regions of parameter space that correspond to the specific and arbitrary ordering of the training examples[1]. An important consequence of this phenomenon, is that even in the presence of a very large (effectively infinite) amount of supervised data, stochastic gradient descent is subject to a degree of *overfitting* to the training data presented early in the training process. In that sense, pre-training is a regularizer which interacts intimately with the optimization process, and when the number of training examples becomes large, its positive effect is seen not only on generalization error but also on training error.

3 Summary of Results

In a first set of experiments, we establish the effect of pre-training on improving the generalization error of trained deep architectures. In the same set we also exploit dimensionality reduction techniques to illustrate how pre-training affects the location of minima in parameter space. In the second set of experiments we directly compare the two alternative hypotheses (pre-training as a pre-conditioner; and pre-training as an optimization scheme) against the hypothesis that pre-training is a regularization strategy. In a final set of experiments, we explore the role of pre-training in the online learning setting, where the number of available training examples grows very large. In these experiments, we test key aspects of our hypothesis relating to the topology of the cost function and the role of pre-training in manipulating the region of parameter space from which supervised training is initiated.

Our results show that pre-training adds robustness to a deep architecture. They also suggest that increasing the depth of an architecture that is not pre-trained increases the probability of finding poor apparent local minima. Pre-trained networks give consistently better generalization. Our visualizations point to the observations that pre-trained networks learn qualitatively different features (if networks are visualized in the weight space) compared to networks without pre-training. Moreover, the trajectories of networks with different initialization seeds seem to fall into many distinct apparent local minima, which are again different (and seemingly far apart) depending on whether we use pre-training or not.

We have shown that pre-training is not simply a way of getting a good initial marginal distribution, and that it captures more intricate dependencies between parameters. One of our findings is that deep networks with pre-training seem to exhibit some properties of a regularizer: with small enough layers, pre-trained deep architectures are systematically worse that randomly initialized deep architectures. Moreover, when the layers are big enough, the pre-trained models obtain worse training errors, but better generalization performance. Additionally, we have re-done an experiment which purportedly showed that pre-training can be explained with an optimization hypothesis and observed a regularization effect instead. We also showed that classical regularization techniques (such as L_1/L_2 penalties on the network weights) cannot achieve the same performance as pre-training, and that the effect of pre-training does not go away with more training data, so if pre-training is a regularizer, it is certainly of a rather different kind.

The two pre-training strategies considered—denoising auto-encoders and Restricted Boltzmann Machines—seem to produce qualitative similar observations. We have observed that, surprisingly, the pre-training advantage is present even in the case of really large training sets, pointing towards the conclusion that the starting point in the non-convex optimization problem is indeed quite important; a fact confirmed by our visualizations of filters at various levels in the network. Finally, the other important set of results show us that pre-training acts like a variance reduction technique, yet a network with pre-training has a lower training error on a very large dataset, which supports an optimization interpretation of the effect of pre-training.

How do we make sense of all these results? We draw a hypothesis, about the dynamics of learning in an architecture that is trained using two phases (pre-training and supervised fine-tuning), which we believe to be consistent with all the above results. This hypothesis suggests that there are consequences of the non-convexity of the supervised objective function, which we observed in various ways throughout our experiments. One of these consequences is that early examples have a big influence on the outcome of training and this is one of the reasons why in a large-scale setting the influence of pre-training is still present. Throughout this paper, we have delved on the idea that the basin of attraction induced by the early examples (in conjunction with pre-training) is, for all purposes, a basin from which supervised training does not escape.

[1] This process seems similar to the "critical period" phenomena observed in neuroscience and psychology (Bornstein, 1987)

This effect can be observed from the various visualizations and performance evaluations that we made. *Pre-training,* **as a regularizer** *that only influences the starting point of supervised training, has an effect that, contrary to classical regularizers, does not disappear* (at least as far as we can show from our results). Basically, unsupervised pre-training favors hidden units that compute features of the input X that correspond to major factors of variation in the true $P(X)$. Assuming that some of these are near features useful at predicting variations in Y, unsupervised pre-training sets up the parameters near a solution of low predictive generalization error.

One of the main messages that our results imply is that the optimization of a non-convex objective function with stochastic gradient descent presents challenges for analysis, especially in a regime with large amounts of data. Our analysis so far shows that it is possible for networks that are trained in such a regime to be influenced more by early examples. This can pose problems in scenarios where we would like our networks to be able to capture more of the information in later examples, i.e., when training from very large datasets and trying to capture a lot of information from them.

One interesting realization is that with a small training set, we do not usually put a lot of importance on minimizing the training error, because overfitting is a major issue; the training error is not a good way to distinguish between the generalization performance of two models. In that setting, pre-training helps to find apparent local minima that have better generalization error. With a large training set, our results suggest that the empirical and true distributions converge. In such a scenario, *finding a better apparent local minimum will matter and stronger (better) optimization strategies should have a significant impact on generalization when the training set is very large*.

Our results suggest that optimization in deep networks is a complicated problem that is influenced in part by the early examples during training. Future work should clarify this hypothesis. If it is true and we want our learners to capture really complicated distributions from very large training sets, it may mean that we should consider learning algorithms that reduce the effect of the early examples, allowing parameters to escape from the attractors in which current learning dynamics get stuck.

Understanding and improving deep architectures remains a challenge. Our conviction is that devising improved strategies for learning in deep architectures requires a more profound understanding of the difficulties that we face with them. This work helps with such understanding via extensive simulations and puts forward and confirms a hypothesis explaining the mechanisms behind the effect of unsupervised pre-training for the final discriminant learning task.

References

Amari, S., Murata, N., Müller, K.-R., Finke, M., & Yang, H. H. (1997). Asymptotic statistical theory of overtraining and cross-validation. *IEEE Transactions on Neural Networks*, *8*, 985–996.

Barron, A. E. (1991). Complexity regularization with application to artificial neural networks. In G. Roussas (Ed.), *Nonparametric functional estimation and related topics*, 561–576. Kluwer Academic Publishers.

Bengio, Y., Lamblin, P., Popovici, D., & Larochelle, H. (2007). Greedy layer-wise training of deep networks. *Advances in Neural Information Processing Systems 19 (NIPS'06)* (pp. 153–160). MIT Press.

Bengio, Y., & LeCun, Y. (2007). Scaling learning algorithms towards AI. In L. Bottou, O. Chapelle, D. DeCoste and J. Weston (Eds.), *Large scale kernel machines*. MIT Press.

Bornstein, M. H. (1987). *Sensitive periods in development : interdisciplinary perspectives / edited by marc h. bornstein*. Lawrence Erlbaum Associates, Hillsdale, N.J. :.

Hinton, G. E., Osindero, S., & Teh, Y. (2006). A fast learning algorithm for deep belief nets. *Neural Computation*, *18*, 1527–1554.

Larochelle, H., & Bengio, Y. (2008). Classification using discriminative restricted Boltzmann machines. *Proceedings of the Twenty-fifth International Conference on Machine Learning (ICML'08)* (pp. 536–543). Helsinki, Finland: ACM.

Lasserre, J. A., Bishop, C. M., & Minka, T. P. (2006). Principled hybrids of generative and discriminative models. *Proceedings of the Computer Vision and Pattern Recognition Conference (CVPR'06)* (pp. 87–94). Washington, DC, USA: IEEE Computer Society.

Ranzato, M., Poultney, C., Chopra, S., & LeCun, Y. (2007). Efficient learning of sparse representations with an energy-based model. *Advances in Neural Information Processing Systems 19 (NIPS'06)* (pp. 1137–1144). MIT Press.

Sjöberg, J., & Ljung, L. (1995). Overtraining, regularization and searching for a minimum, with application to neural networks. *International Journal of Control*, *62*, 1391–1407.

Vincent, P., Larochelle, H., Bengio, Y., & Manzagol, P.-A. (2008). Extracting and composing robust features with denoising autoencoders. *Proceedings of the Twenty-fifth International Conference on Machine Learning (ICML'08)* (pp. 1096–1103). ACM.

Weston, J., Ratle, F., & Collobert, R. (2008). Deep learning via semi-supervised embedding. *Proceedings of the Twenty-fifth International Conference on Machine Learning (ICML'08)* (pp. 1168–1175). New York, NY, USA: ACM.

Zhu, L., Chen, Y., & Yuille, A. (2009). Unsupervised learning of probabilistic grammar-markov models for object categories. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *31*, 114–128.