ESANN 2014 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 23-25 April 2014, i6doc.com publ., ISBN 978-287419095-7.
Available from http://www.i6doc.com/fr/livre/?GCOI=28001100432440.

A HMM-based Pre-training Approach for Sequential Data

Luca Pasa[1], Alberto Testolin[2], Alessandro Sperduti[1]

1- Department of Mathematics
2- Department of Developmental Psychology and Socialisation
University of Padova - Italy

Abstract. Much recent research highlighted the critical role of unsupervised pre-training to improve the performance of neural network models. However, extensions of those architectures to the temporal domain introduce additional issues, which often prevent to obtain good performance in a reasonable time. We propose a novel approach to pre-train sequential neural networks in which a simpler, approximate distribution generated by a linear model is first used to drive the weights in a better region of the parameter space. After this smooth distribution has been learned, the network is fine-tuned on the more complex real dataset. The benefits of the proposed method are demonstrated on a prediction task using two datasets of polyphonic music, and the general validity of this strategy is shown by applying it to two different recurrent neural network architectures.

1 Introduction

Even if deep learning systems reach state-of-the-art performance in several machine learning tasks, their computational complexity is still a limit in many real-word scenarios. This issue has been partially tackled with the advent of new high performance parallel computing architectures, which exploit powerful graphic processors to speed-up learning algorithms [1]. However, the breakthrough that allowed to effectively train large-scale networks has been the introduction of an unsupervised pre-training phase [2], where the objective is to build a good generative model of the data that can eventually be fine-tuned using a supervised criterion. Pre-training drives the network weights in a region where optimization is somehow easier, thus helping the fine-tuning phase to reach better local optima. It might also act as a regularizer, by introducing a bias towards good configurations of the parameter space [3]. Although the benefits of pre-training have been extensively investigated in the static domain (e.g., learning images encoded as fixed-size vectors), it is not yet clear how this approach should be extended to the temporal domain, where the aim is to model sequences of events. Dealing with time poses many challenges, because temporal dependencies limit the parallelization. Despite recent advances in training recurrent networks [4], improving their convergence speed is therefore still challenging. A possible solution is to pre-train only input-to-hidden connections, thereby ignoring temporal information (encoded by hidden-to-hidden connections) by considering each element of the sequence as independent from the others [5].

In this paper we propose a different pre-training strategy, which is reminiscent of the idea of curriculum learning [6]. The rationale behind this approach is

ESANN 2014 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 23-25 April 2014, i6doc.com publ., ISBN 978-287419095-7.
Available from http://www.i6doc.com/fr/livre/?GCOI=28001100432440.

that complex problems should be learned by starting from simpler concepts and then increasing the difficulty level by gradually showing more complex training examples to the learning agent. To this aim, instead of using the same dataset for both pre-training and fine-tuning, we first use a linear Hidden Markov Model (HMM; [7]) to generate a new dataset, which represents an approximation of the target probability distribution. We use this simpler dataset to pre-train the more powerful non-linear model, which is subsequently fine-tuned on the real data. We tested our method on a complex temporal task, which required to learn the structure of polyphonic music encoded using symbolic sequences (piano-roll MIDI format). We first applied the HMM pre-training on a recently proposed recurrent architecture [5] that has been shown to obtain state-of-the-art performance on a prediction task for the considered dataset, which consisted in predicting the notes that will be played at the next time step given all previously played notes in the sequence. We then assessed the robustness and the generality of the method by applying it also to a classic recurrent neural network (RNN). Our results demonstrate the value of the proposed pre-training strategy, which allows to learn a good model of the data in a significantly shorter time.

2 RNN and RBM-based models for sequential data

RNNs are a popular family of neural networks used to model sequential data. At each time step, the current element is presented to the network through a layer of input units. An output layer is used to predict the next element of the sequence, and a layer of hidden units encodes the latent features of the data. RNNs typically exploit non-linear (e.g., logistic) activation functions, which ensure expressive power while maintaining efficient learning. Formally, given a sequence of input vectors $x_1, x_2, ..., x_T \in \mathbb{R}^n$, the RNN computes the sequence of hidden states $h_1, h_2, ..., h_T \in \mathbb{R}^m$ and the sequence of output states $o_1, o_2, ..., o_T \in \mathbb{R}^k$ by iterating the following equations:

$$\begin{aligned} h_i &= \sigma(W_{hx}x_i + W_{hh}h_{i-1} + b_h) \\ o_i &= \sigma(W_{oh}h_i + b_o) \end{aligned}$$

where σ is the logistic function, b_h and b_o are the biases of hidden and output units, and W_{hx}, W_{oh}, and W_{hh} are the input-to-hidden, hidden-to-output and hidden-to-hidden weights. RNNs can be trained using backpropagation through time, which is a temporal variant of stochastic gradient descent (SGD).

The Recurrent Neural Network - Restricted Boltzmann Machine (RNN-RBM; [5]) is a sequential model that combines an RNN with an RBM. This architecture merges the best feature of RNNs (i.e., the ability to learn temporal dependencies) with that of RBMs (i.e., the ability to efficiently model multimodal distributions by means of energy functions). RNN-RBMs are stochastic models, for which the joint probability distribution of hidden and input units is defined as:

$$P(v^{(t)}, h^{(t)}) = \prod_{t=1}^{T} P(v^{(t)}, h^{(t)} | v^{(t-1)}, v^{(t-2)}, ..., v^{(1)}, \hat{h}^{(t-1)}, \hat{h}^{(t-2)}, ..., \hat{h}^{(1)})$$

ESANN 2014 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 23-25 April 2014, i6doc.com publ., ISBN 978-287419095-7.
Available from http://www.i6doc.com/fr/livre/?GCOI=28001100432440.

where: $\hat{h}^{(t)} = \sigma(W_2 v^{(t)} + W_3 \hat{h}^{(t-1)} + b_{\hat{h}})$ and $v^{(t)}$, $h^{(t)}$ and $\hat{h}^{(t)}$ represent, respectively, the input units, the RBM-hidden units and RNN-hidden units, whereas $b_{\hat{h}}$ represents RNN-hidden unit biases (see Fig. 1). The authors of the RNN-RBM demonstrated the importance of the pre-training phase in order to obtain a good model of the data. In particular, their original approach consisted in separately pre-train the RBM-part of the model using the contrastive divergence algorithm [2], and then the RNN-part of the model using either the SGD method or a recently proposed Hessian-Free (HF) optimization method [4].

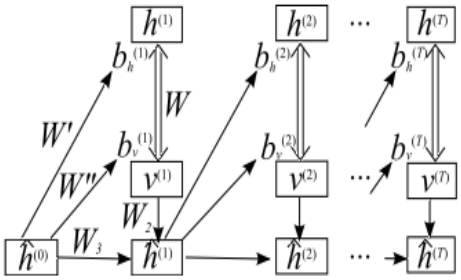

Fig. 1: Schematic representation of the RNN-RBM (see [5] for details).

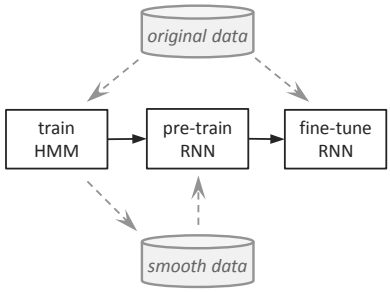

Fig. 2: Flow chart of the proposed HMM-based pre-training method.

3 HMM-based pre-training

The pre-training method that we propose improves the learning process by first modeling an approximation of the real probability distribution of the data. This initial, imprecise knowledge will serve as the basis for the fine-tuning phase, which benefits from this better starting point both in terms of convergence speed as well as in terms of the quality of the final model (i.e., higher prediction accuracy). The idea is to first learn a smoothed version of the data and gradually consider less smoothing, with the intuition that a smooth version of the problem reveals the global picture and therefore allows improving during the fine-tuning phase. To this aim, a simpler probabilistic model is first trained on the real dataset. We chose to use HMMs because they represent a sound, powerful and efficient framework for modeling sequential data, and despite their linear formulation they reach state-of-the-art results in many practical problems, including music modeling [5]. Once the HMM has learned a model of the data, we used it to generate a fixed number of sequences that populated a new dataset, which constituted an approximated version of the real polyphonic sequences. These simplified patterns are then used to pre-train the non-linear model, with the aim of transferring the knowledge acquired by the HMM to the more powerful recurrent network. The number of sequences generated by the HMM is an important parameter that can be chosen according to the dimension of the training set. In our study, 500 sequences were enough to obtain good performances.

ESANN 2014 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 23-25 April 2014, i6doc.com publ., ISBN 978-287419095-7.
Available from http://www.i6doc.com/fr/livre/?GCOI=28001100432440.

After pre-training, the non-linear model is fine-tuned on the real dataset in order to refine the distribution learned during pre-training. A flow chart of the HMM-based pre-training method is given in Fig. 2.

4 Experimental results

We tested our pre-training method on both the recurrent architectures described above, using two different datasets. The first one (Nottingham) contains folk songs, which have a small number of different chords and redundant structure (mean sequences length: 212; max length: 1491; min length: 35; sequences in training set: 694; sequences in test set: 170). The second one (Piano-midi.de) is an archive of classic piano music, which contains more complex songs with many different chords (mean sequences length: 812; max length: 4405; min length: 78; sequences in training set: 87; sequences in test set: 25). Learning the structure of polyphonic music with HMMs was challenging due to the exponential number of possible configurations of notes that can be produced at each time step, which would cause the alphabet of the model to have an intractable size. We solved this issue by only considering the configurations that were actually present in the dataset, which reduced the complexity of the alphabet but at the same time maintained enough variability to produce realistic samples. We assessed the accuracy of the models on the prediction task defined in [5], using the same evaluation metric and model parameters. We also collected the total training times, which were affected by both the pre-training phase and the convergence speed of the fine-tuning phase.

For the RNN-RBM, we compared our pre-training method with those used by the authors of the model. The RBM- and RNN-part of the model had, respectively, 150 and 100 hidden units. Pre-training was performed for 100 epochs, and fine-tuning for 200 epochs. Total training times and prediction accuracies for the HMM, SGD and HF pre-trainings are reported in Fig. 3. In general, different pre-training methods led to similar accuracies at the end of the fine-tuning phase. However, in the more complex Piano-midi dataset our HMM pre-training obtained slightly better results. Regarding convergence speed, the HMM method always significantly outperformed the others (e.g., it saved more than 8 hours of computing in the Nottingham dataset). We also assessed the change in performance as the number of HMM states varies. As expected, using a smaller number of hidden states (≤ 25) reduced pre-training times. Interestingly, this did not affect the quality of the models after the fine-tuning phase, which still converged to good solutions. Using a HMM with 50 states, instead, was detrimental due to the slow convergence speed of the HMM training. Finally, the HMM pre-training seemed to perform a better initialization of the network, which allowed to improve convergence speed also during the fine-tuning phase. For example, the network pre-trained with the HMM reached the highest accuracy after only 110/120 epochs, compared to 200 epochs required by the other methods. It is worth noting that the accuracies measured directly on the HMMs were always fairly low, at best approaching 53% in the Nottingham dataset and

ESANN 2014 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 23-25 April 2014, i6doc.com publ., ISBN 978-287419095-7.
Available from http://www.i6doc.com/fr/livre/?GCOI=28001100432440.

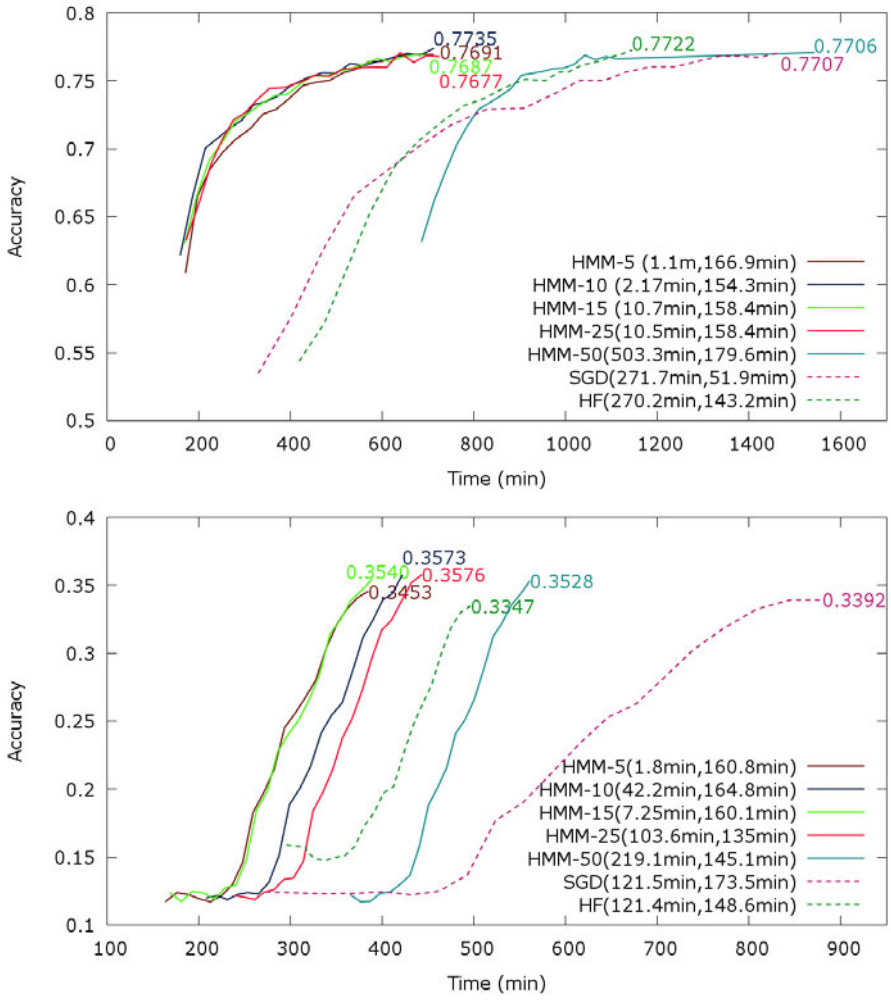

Fig. 3: Accuracy and running times of the tested pre-training methods, measured on the Nottingham (top) and Piano-midi (bottom) datasets. In parentheses: times for training the HMM and for pre-training the network (our method) and for RBM and RNN pre-training (SGD and HF methods).

10.1% in the Piano-midi dataset. For the simpler RNN, we compared the final accuracy obtained using the HMM method with that obtained in [5] and, as a baseline, with an RNN without pre-training. The implemented RNN had 200 hidden units, pre-training was performed for 400 epochs and fine-tuning for 800 epochs. The benefits of introducing the pre-training are clearly shown by the marked difference on prediction accuracy (Nottingham dataset): 10,1% for the network without pre-training, 66.64% with the pre-training used in [5] and 69,5%

ESANN 2014 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 23-25 April 2014, i6doc.com publ., ISBN 978-287419095-7.
Available from http://www.i6doc.com/fr/livre/?GCOI=28001100432440.

with the HMM pre-training. Notably, using the HMM method the accuracy of the simple RNN approaches that of the more complex RNN-RBM.

5 Conclusions and future directions

In this paper we proposed a novel method for pre-training recurrent neural networks, which consists of generating a smoothed, approximated dataset using a linear HMM trained with the original data. When applied to a recently proposed recurrent neural network architecture [5], our HMM-based pre-training led to prediction accuracies comparable (and sometimes greater) than those obtained with currently available pre-training strategies, but requiring a significantly lower computational time. We also tested the method on a basic recurrent neural network, and also in this case the effectiveness of our approach was confirmed. It should be stressed that the proposed method does not need *ad-hoc* adjustments of existing learning algorithms, because it consists in generating a simplified dataset that will be used to initialize the parameters of the learner. Our pre-training strategy is therefore very general, and its benefits could be readily extended to pre-train many other types of sequential models.

Even if our results are encouraging, further research is needed to better characterize the strengths (and possibly the weaknesses) of the proposed approach. In particular, a formal characterization of the method could improve its applicability in other domains. Indeed, the method should be evaluated on different datasets and tasks, in order to verify whether the high performance is maintained even if the objective does not measures mean prediction accuracies, which are likely to be easily captured by linear models like HMMs. Finally, future experiments should better explore the parameters tuning, because there could still be room for improvement by carefully selecting the number of epochs, number and length of sequences generated by the HMM, network parameters (e.g., number of hidden units) and other meta-parameters of the learning algorithms.

References

[1] R. Raina, A. Madhavan, and A.Y. Ng. Large-scale deep unsupervised learning using graphics processors. *International Conference on Machine Learning*, pages 110–880, 2009.

[2] G.E. Hinton and R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–7, 2006.

[3] D. Erhan, Y. Bengio, and A. Courville. Why does unsupervised pre-training help deep learning? *The Journal of Machine Learning Research*, 11:625–660, 2010.

[4] J. Martens and I. Sutskever. Learning recurrent neural networks with Hessian-free optimization. *International Conference on Machine Learning*, pages 1033–1040, 2011.

[5] N. Boulanger-Lewandowski, Y. Bengio, and V. Pascal. Modeling temporal dependencies in high-dimensional sequences: Application to polyphonic music generation and transcription. *arXiv preprint arXiv:1206.6392*, 2012.

[6] Y. Bengio, J. Louradour, R. Collobert, and J. Weston. Curriculum learning. *International Conference on Machine Learning*, pages 1–8, 2009.

[7] L.R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.