IMPROVED PRE-TRAINING OF DEEP BELIEF NETWORKS USING SPARSE ENCODING SYMMETRIC MACHINES

Christian Plahl[1]*, Tara N. Sainath*[2]*, Bhuvana Ramabhadran*[2]*, and David Nahamoo*[2]

[1]Lehrstuhl für Informatik 6 - Computer Science Department
RWTH Aachen University, Aachen, Germany
[2]IBM T. J. Watson Research Center, Yorktown Heights, NY 10598
plahl@cs.rwth-aachen.de[1], {tsainath, bhuvana, nahamoo}@us.ibm.com[2]

ABSTRACT

Restricted Boltzmann Machines (RBM) continue to be a popular methodology to pre-train weights of Deep Belief Networks (DBNs). However, the RBM objective function cannot be maximized directly. Therefore, it is not clear what function to monitor when deciding to stop the training, leading to a challenge in managing the computational costs. The Sparse Encoding Symmetric Machine (SESM) has been suggested as an alternative method for pre-training. By placing a sparseness term on the NN output codebook, SESM allows the objective function to be optimized directly and reliably be monitored as an indicator to stop the training. In this paper, we explore SESM to pre-train DBNs and apply this the first time to speech recognition. First, we provide a detailed analysis comparing the behavior of SESM and RBM. Second, we compare the performance of SESM pre-trained and RBM pre-trained DBNs on TIMIT and a 50 hour English Broadcast News task. Results indicate that pre-trained DBNs using SESM and RBMs achieve comparable performance and outperform randomly initialized DBNs with SESM providing a much easier stopping criterion relative to RBM.

Index Terms— Deep belief network, pre-training, neural network feature extraction, sparse representation

1. INTRODUCTION

In recent years, neural networks (NN) have become a major component of state-of-the-art speech recognition systems [1]. The conventional approach to train such NN is limited to few hidden layers. Since the weights of the NN training are initialized randomly and the objective function is non-convex, the trained weights tend to get stuck in a poor local optimum. Recently, [2] has introduced an unsupervised generative method to pre-train the weights of NNs using Restricted Boltzmann Machines (RBM). A NN which is pre-trained layer by layer with RBMs is generally referred as deep belief network (DBN). The main idea to pre-train NNs is to provide a good initialization of the weights and to include regularization in the training.

The concept of pre-training has been successfully adapted to the speech recognition task as shown in [3, 4, 5]. Nevertheless, the main issue with RBM is that the objective function cannot be maximized directly. Instead, the concept of contrastive divergence (CD) is applied to approximate the objective function [6]. Therefore it is unclear which function to monitor to obtain a good training stopping criteria, resulting in potentially high computational costs to pre-train the weights of a NN. Sparse Encoding Symmetric Machines (SESM) has been explored for pre-training of handwritten digits [7], but not applied to any speech recognition task yet. Instead of using CD, the objective function is maximized directly by SESM. In this paper, we will show that SESM are competitive to RBMs and that SESMs provide a clear stopping criteria, whereas for RBM no such clear criteria exists.

Our experiments are conducted on both TIMIT and a 50 hours English broadcast news task. We use these corpora to analyze pre-training using RBMs and SESMs, including sparsity of the output and the weight correlation before and after pre-training. In addition, we show that DBNs pre-trained using SESM and RBMs achieve comparable performance and outperform randomly initialized DBNs, although SESMs are easier to train as they provide a much easier stopping criterion when compared to RBM.

2. PRE-TRAINING OF NEURAL NETWORKS

In practice, training of the entire network at once becomes difficult as the number of hidden layers increases and the weight connections are initialized randomly. Instead, by training one layer at a time and each layer separately, the training of multiple layers is successful [2]. This concept is used to pre-train DBN, where each layer is trained in an unsupervised manner. Instead of using a random initialization of the weights in the NN, training the weights layer by layer is an efficient and effective method to find a good starting point for the weights of a huge NN.

After training the weights in an unsupervised manner, the different layers are stacked together and a supervised fine-tuning step is applied. The fine-tuning step is the conventional training approach of a NN using back propagation but with previously trained weights as the starting point.

In the pre-training step, the unsupervised model is defined as a distribution over the input vector Y through an energy function $E(Y, Z, W)$:

$$P(Y|W) = \int_z P(Y, z|W) = \frac{\int_z e^{-\beta E(Y,z,W)}}{\int_{y,z} e^{-\beta E(y,z,W)}}, \qquad (1)$$

where β is an arbitrary constant. The weight matrix W is updated during training to obtain the (optimal) code Z. Finding the weights of the NN results in minimizing a loss function which is equal to the negative log likelihood of the training data.

$$L(W, Y) = -\frac{1}{\beta} \int_z e^{-\beta E(Y,z,W)} + \frac{1}{\beta} \int_{y,z} e^{-\beta E(y,z,W)} \qquad (2)$$

Here, the first term in Equation 2 is the free energy, the second term the so called *log partition function*. The log partition function ensures that we observe low energy for the (true) data distribution and

high values anywhere else [7]. The main issue is that the gradient of the log partition function w.r.t. the parameters W could be hard to estimate.

In the following sections, we will briefly describe two approaches to approximate the log partition function. Whereas RBMs use the concept of contrastive divergence (CD) to obtain the log partition function, SESMs replace the log partition term by adding sparseness on the output of the neural network. RBMs have been successfully explored for speech recognition [4, 5], but SESMs have not yet been applied to any speech recognition tasks.

2.1. Restricted Boltzmann Machine

Restricted Boltzmann Machines are a effective way to initialize the weights of a unsupervised trained DBN [2, 4, 5]. In order to optimize the parameters the reconstruction of the input Y is taken into account. Therefore, we distinguish the encoder step, which consists of the forward step of the NN, and the decoder step, where the output is reconstructed from the encoder.

In addition, depending on the distribution for the visible and hidden layer we distinguish a Gaussian and a Bernoulli distribution. The energy function, including the encoder and decoder part, is described by:

$$E(y, z, W) = -Z^T W^T Y - U - b_{enc}^T Z, \quad (3)$$

where $U = \frac{1}{2}(b_{dec}^T Y)^2$ for the Gaussian-Bernoulli distribution and $U = b_{dec}^T Y$ for Bernoulli-Bernoulli. The bias terms for encoding b_{enc} and decoding b_{dec} as well as the weights W are trained using contrastive divergence (CD) [6]. In order to estimate the log partition function, the main idea of CD is to create the output Z by sampling and reconstruct the input by using the sampled Z.

2.2. Sparse Encoding Symmetric Machine

In contrast to RBMs, Sparse Encoding Symmetric Machines do not rely on an explicit contrastive term in the loss function [7]. The log partition function is replaced by a sparseness penalty term on the output obtained by the encoder. The sparseness term allows the direct optimization of the objective function. The training of SESMs is performed by simply minimizing the average energy in combination with the additional sparseness term of the output. Similar to the RBM, SESM follows the encoder-decoder paradigm. The encoder and decoder are described by:

$$f_{enc}(Y) = W^T Y + b_{enc}, \quad f_{dec}(Z) = Wl(Z) + b_{dec} \quad (4)$$

where the function l is a point-wise logistic non-linearity of the form: $l(x) = 1/(1 + exp(-gx))$ with a fixed gain $g = 1$ for all our experiments.

The free energy in Eqn. 1 and Eqn. 2 of the SESM is described by

$$E(Y, Z) = \alpha_e ||Z - f_{enc}(Y)||^2 + ||Y - f_{dec}(Z)||^2. \quad (5)$$

The free energy is divided into the difference between the current observed code f_{enc} and its optimal solution Z, scaled by a constant $\alpha_e = 1$, and the difference of input Y and its reconstruction f_{dec}.

Overall, we optimize the following loss function during training, obtained form Equation 2 and 5:

$$\begin{aligned} L(W) &= E(Y, Z) + \alpha_s h(Z) + \alpha_r |W|_1 \\ &= \alpha_e ||Z - f_{enc}(Y)||_2^2 + |Y - f_{dec}(Z)|_2^2 \\ &\quad + \alpha_s h(Z) + \alpha_r |W|_1, \end{aligned} \quad (6)$$

where $h(Z) = \sum_d \log(1 + l^2(z_d))$. The loss contains the free energy (Eqn. 5), a sparseness term ($h(Z)$) as an approximation to the log partition function and a l_1-regularization term on the weights. Instead of sampling the output as for RBM, SESM uses the output of the encoder directly.

2.3. Sparse Encoding Symmetric Machine Training Recipe

We have modified the proposed training procedure for SESM by [7] and the pre-training concept by [2]. The optimization of the weights W and the code Z is difficult, so we use an iterative procedure to first estimate the optimal code Z^* and afterwards the updates of W, b_{enc} and b_{dec}. In order to find the best solution the gradient decent algorithm is applied. For pre-training, we have used the following algorithm, where *loss* is the loss over all training samples and *lossZ* is the loss for the current sample/utterance using fixed weights W:

WHILE $\Delta loss \geq 0$
  FOR each training sample/utterance
    Compute initial value of optimal code (Z^*)
    WHILE $\Delta lossZ \geq threshold$
      Get gradient w.r.t. Z *and update code* Z
      Anneal leaning rate if $\Delta lossZ \leq 0$
    Get gradient w.r.t. W *and update parameter* W, b_{enc} *and* b_{dec}
  IF $loss \leq 0$*: anneal leaning rate* η
  IF fixed # of iterations OR # of anneals of η *reached: STOP*

Depending on the layer to be trained we applied the following rules to set the learning rate η and the sparseness parameter α_s:

Layer-1: Choose a high value for α_s to obtain a sparse output and use a high learning rate η to achieve a lot of structure in the pre-trained weights. In our experiments we set $\alpha_s = 0.2$ and $\eta = 0.005$.

Layer-n: The output should be less sparse compared to the previous layer (current input). We decreased α_s by a factor of 2 to 4, also depending on the increase/decrease of the layer size. The learning rate η has to be updated as well. Due to lower sparseness in the output, a lower learning rate is required. We decreased the learning rate by a magnitude or more.

3. EXPERIMENTAL SETUP

The experiments are performed on a small and a large vocabulary task. First, the small vocabulary recognition experiments are conducted on TIMIT [8] and results are reported on its core test set. Large vocabulary experiments are conducted on a English broadcast news (BN) transcription task. Models are trained on 50 hours of data from the 1996 + 1997 English Broadcast News Speech Corpora and results are reported on 3 hours of the EARS Dev-04f set.

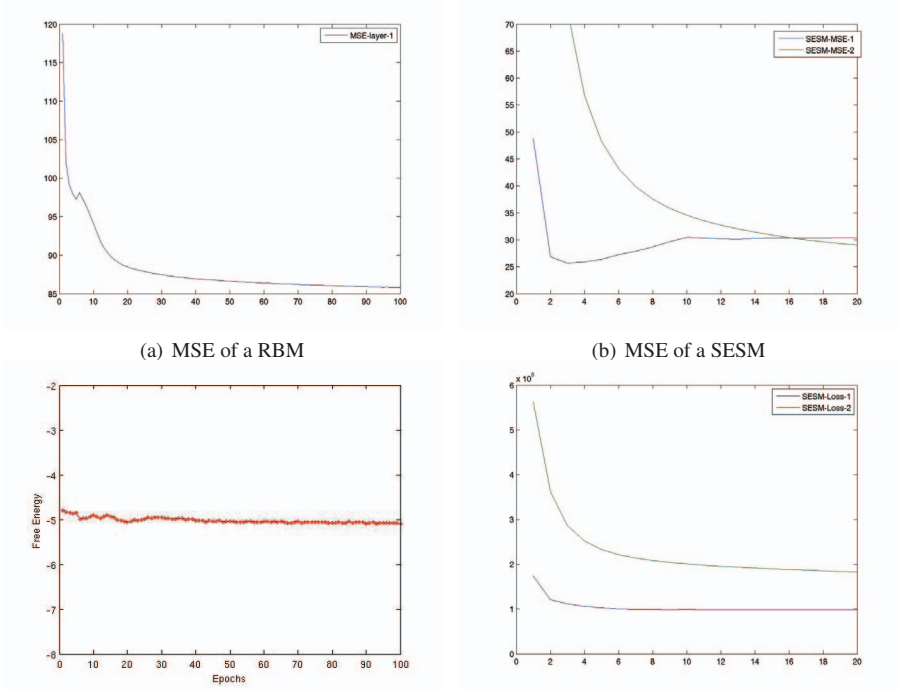
For training and testing the NN on TIMIT we have created a set of discriminatively trained features using the boosted Maximum Mutual Information (BMMI) criterion. For BN, vocal tract length normalized cepstral features are used. In the fine tuning step the NNs are trained on the context dependent triphone states, observed by a previously trained HMM model. In recognition, the posteriors derived from the NN are used for decoding rather than as features for a HMM model. All experiments are performed using the IBM speech recognition toolkit [9].

4. RESULTS

We ran several experiments keeping the starting point of the weights for pre-training of RBM and SESM the same. First we present a detailed analysis performed on TIMIT. Next, we demonstrate the competitive performance of RBM and SESM using TIMIT and BN.

4.1. Stopping Criteria

As mentioned above, the main issue of the RBM training is that a clear criteria to stop the pre-training is missing. Typically the mean squared error (MSE) between the input an its reconstruction, an indicator how well the reconstruction is done, is used to measure the training performance. As shown in Figure 1(a) the MSE is decreased over all iterations. Whereas the MSE is not correlated to the loss, the free energy is, see Eqn. 2. Since the free energy could not be



(a) MSE of a RBM

(b) MSE of a SESM

(c) Approximation to the free energy of a RBM

(d) Loss of a SESM

Fig. 1. Development of the MSE for RBM (a) and SESM (b) and the approximated free energy for RBM (c) and the loss of SESM (d).

estimated directly, we use an approximation. Nevertheless, the difference in free energy between consecutive iterations of an RBM is decreasing over all iterations as shown in Figure 1(c). Even when the MSE and the loss of an RBM seem to behave similarly, no clear correlation is observed. In contrast, the loss of a SESM is optimized directly. Depending on the parameters used, the MSE decreases monotonically if the sparseness value and learning rate are set very low —green curve in Figure 1(b) and (d) Using the recipe presented in Section 2.3, we obtain a different behavior. During training, the MSE gets worse, while the overall loss is decreased (blue curve). The most important and dominant term in the loss function of the SESM belongs to the MSE term, see Equation 6. When the MSE is very low, sparseness becomes more significant resulting in a sparse output. Even though MSE increases, the overall loss still decreases as illustrated in Figure 1(d). Here, the MSE and the loss start to increase and after 3 and 8 iterations respectively. After annealing the leaning rate the loss is decreased slightly, whereas the MSE increases further and we obtain more sparseness on the output.

Due to the interaction of sparseness and MSE term in the objective function, Figure 1(b) indicates that the MSE is not the optimal criteria to stop the training of SESMs. Furthermore, we have not observed any improvements compared to the random initialization, when a fine-tuning step is performed using the weights with lowest MSE (results not reported).

Overall, the change in the objective function and adaptation of the leaning rate are more reliable as stopping criteria for training. In our experiments on TIMIT, the stopping criteria was reached after 14 iterations when the final number of annealing steps was reached. For RBMs no such indicator exists. We have observed subsequently that if we decrease the number of iterations to train an RBM to match the SESM, the final WER after fine-tuning for both RBM and SESM is the same. However, on the BN task, we have observed a 0.6% absolute difference —from 27.1% to 26.5%— even though the number of training iterations are doubled for the RBM.

4.2. Sparseness and Weight Correlation

In addition to the previous experiments, we analyzed the sparseness of the output produced by the SESM and the correlation between the weight matrix before and after fine-tuning.

In order to maximize the objective function of a SESM directly, a sparseness term is added to the loss, resulting in a sparse output —robust against noise in the data— rather than a sparse representation of the weights. Sparse weights seems to be harmful in early training iterations of an RBM [4]. While we can control the sparseness of the code produced by a SESM, RBMs do not have an explicit sparseness term in the objective function. We have calculated the sparseness of

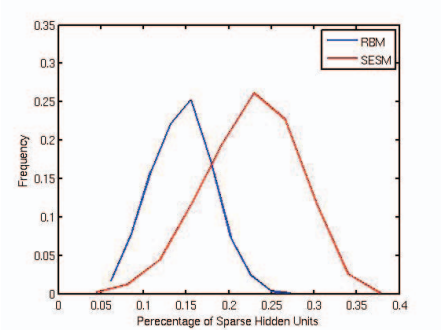
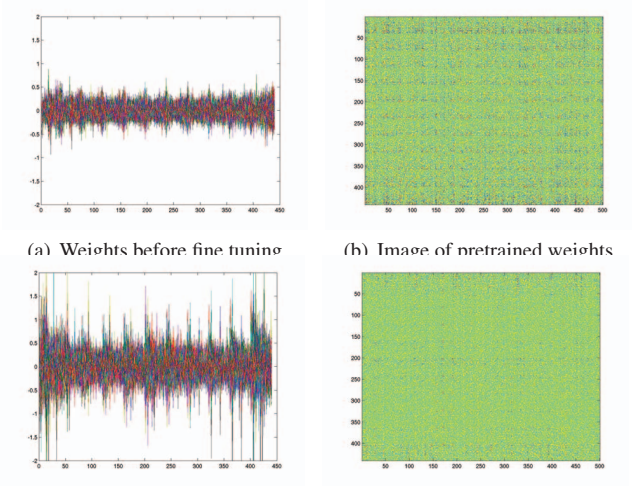


Fig. 2. Output activation for RBM and SESM.

the output activation of the first layer of an RBM and a SESM over 250k frames. As shown in Figure 2 the output activations of the first layer of a SESM are much sparser than the output activations of an RBM, where no sparseness term exists. As expected, SESM contains only a few high activations, while the RBMs contain many more high activations. Moreover, when we take all values lower than 0.001 into account, the sparseness of an RBM is around 14.6% and for SESM 22.7%.



(a) Weights before fine tuning (b) Image of pretrained weights

(c) Weights after fine tuning (d) Image of the final weight matrix

Fig. 3. Weights of layer-1 pre-training by SESM, (a) and (b), and after fine tuning, (c) and (d).

Furthermore, as shown in Figure 3, the pre-training of the weights discovers a hidden structure in the data and peaks are localized. After fine-tuning a high correlation between the pre-trained and fine-tuned weights exists. Moreover, the localized high weight activations are been enlarged, to distinguish the target classes, whereas the global weight structure is kept.

Table 1. Phonetic error results (PER) after fine-tuning of a 4 layer network on TIMIT. The results are for random weights initialization, pre-training by RBMs and SESMs.

NN weight initialization [%]		
Random	RBM	SESM
20.7	19.3	19.1

4.3. Experiments on Broadcast News

The final results for TIMIT using the recipe described in Section 2.3 are presented in Table 1. We extended the application of SESM to the BN task, keeping in mind the lessons learned from TIMIT. Table 2 illustrated the results obtained on the BN task using random initialization and pre-training via RBMs and SESMs. As shown in [5] for several feature sets on the same task, DBNs improve over a GMM/HMM system. While the RBM is 0.2% absolute better in performance when compared to the SESM, both forms of pre-training outperform random initialization. Overall, we obtained a relative improvement of 8% on the TIMIT task and 3% on the BN task.

Table 2. Word error results (WER) after fine-tuning of a 4 layer network on BN. The results are for random weights initialization, pre-training by RBMs and SESMs.

NN weight initialization [%]		
Random	RBM	SESM
26.5	25.6	25.8

Further improvements can be obtained by mixing RBM and SESM pre-training. We are able to improve over the best 3-layer RBM result of 27.1% [5] by 0.3% absolute to a final WER of 26.9% by pre training the first layer via a SESM and the second via a RBM, which benefits from the sparse output obtained from the SESM. When the first two layers are trained via SESM followed by a third layer that is trained via a RBM, we do not observe any improvements over the best results. Further investigation are necessary to uncover the best configuration.

5. SUMMARY AND CONCLUSION

In this paper, we examined an alternative and competitive method to pre-train weights of a neural network for initialization the weights for a final NN training pass using back propagation. While the RBM relies on CD, SESM directly optimizes the objective function. Moreover, SESM provides an easy criteria to stop the training, leading to fewer iterations. Independently of using RBM or SESM, we achieve up to 8% relative in PER on TIMIT and up to 3% relative in WER on BN by pre-training. By mixing the two approaches we improve over the best 3-layer based system even further by 0.3% WER absolute.

We also performed a detailed analysis of the weights before and after pre-training and examined the sparseness of the output of SESM and RBM. We showed that during fine-tuning, structure of the pre-trained weights is preserved while the dynamic range of the weights is enlarged. As the sparseness of the SESM is directly controlled and can be optimized, it can be tuned to directly impact WER.

Further investigations are necessary to find the best combination of RBM and SESM and to establish the sensitivity to the pre-training of the first layer. Additionally, a fine-tuning step after each layer could be useful.

6. ACKNOWLEDGEMENT

Thank you to Marc'Aurelio Ranzato for useful discussions related to SESMs. This work was partly realized as part of the Quaero Programme, funded by OSEO, French State agency for innovation.

7. REFERENCES

[1] M. Sundermeyer et.al, "The RWTH 2010 quaero ASR evaluation system for English, French, and German," in *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing*, Prague, Czech Republic, May 2011, pp. 2212–2215.

[2] Geoffrey E. Hinton, Simon Osindero, and Yee-Whye Teh, "A fast learning algorithm for deep belief nets," *Neural Computation*, vol. 18, no. 7, pp. 1527–1554, July 2006.

[3] Abdel-rahman Mohamed, Dong Yu, and Li Deng, "Investigation of full-sequence training of deep belief networks for speech recognition," in *Interspeech*, Makuhari, Japan, Sept. 2010, pp. 1692–1695.

[4] Frank Seide, Li Gang, and Yu Dong, "Conversational Speech Transcription using context-dependent Deep Neural Network," in *Interspeech*, Florence, Italy, Aug. 2011, pp. 437–440.

[5] Tara N. Sainath, Brian Kingsbury, Bhuvana Ramabhadran, Petr Fousek, and P. Novak, "Making Deep Belief Networks effective for Large Vocabulary Continuous Speech Recognition," in *Proc. IEEE Automatic Speech Recognition and Understanding Workshop*, Hawaii, USA, Dec. 2011.

[6] Geoffrey E. Hinton, "Training products of experts by minimizing contrastive divergence," *Neural Computations*, vol. 14, no. 8, pp. 1771–1800, Aug. 2002.

[7] Marc'Aurelio Ranzato, Y-Lan Boureau, and Yann LeCun, "Sparse feature learning for deep belief networks," in *Advances in Neural Information Processing Systems*, 2007.

[8] Lori F. Lamel, Robert H. Kassel, and Stephanie Seneff, "Speech database development: Design and analysis of the acoustic-phonetic corpus," in *Proceedings of the DARPA Speech Recognition Workshop*, 1986, pp. 100–110.

[9] Hagen Soltau, Georg Soan, and Brian Kingsbury, "The IBM Atilla Speech Recognition Toolkit," in *IEEE Workshop on Spoken Language Technology*, Berkley, CA, USA, May 2010, pp. 97–102.