Sparse Auto-associative Neural Networks: Theory and Application to Speech Recognition

G.S.V.S. Sivaram, Sriram Ganapathy, Hynek Hermansky

The Center for Language and Speech Processing,
Human Language Technology Center of Excellence,
The Johns Hopkins University, USA.
e-mail : {*sivaram, ganapathy, hynek*}*@jhu.edu*

Abstract

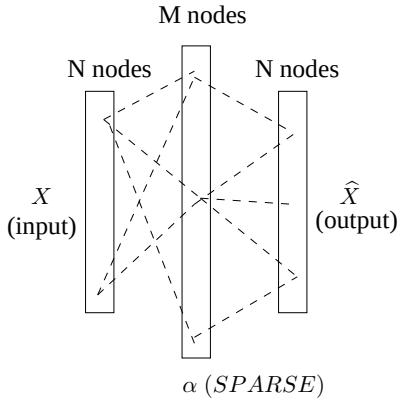
This paper introduces the sparse auto-associative neural network (SAANN) in which the internal hidden layer output is forced to be sparse. This is achieved by adding a sparse regularization term to the original reconstruction error cost function, and updating the parameters of the network to minimize the overall cost. We show applicability of this network to phoneme recognition by extracting sparse hidden layer outputs (used as features) from a network which is trained using perceptual linear prediction (PLP) cepstral coefficients in an unsupervised manner. Experiments with the SAANN features on a state-of-the-art TIMIT phoneme recognition system show a relative improvement in phoneme error rate of 5.1% over the baseline PLP features.

Index Terms: auto-associative neural networks, sparsity, feature extraction, phoneme recognition.

1. Introduction

Capturing the underlying structure of multivariate data in some representation is desirable for many applications such as pattern classification, regression and data visualization. Conventional unsupervised learning methods such as principal component analysis (PCA) and auto-associative neural networks (AANN) [1, 2] rely on reducing the dimensionality of the representation while preserving variance of the data. Recent unsupervised sparse methods like non-negative matrix factorization (NMF) [3], sparse encoding symmetric machine (SESM) [4] and sparse deep belief nets [5] force representations to be sparse when describing the data.

We propose to enforce sparsity in the hidden layer of an AANN. AANN is a feedforward neural network with equal number of input and output nodes, and consisting of a hidden bottleneck layer (can be linear or nonlinear) which has fewer nodes than the input or output layers. The network parameters (weights and biases at each layer) are adjusted using the back-propagation algorithm to learn an identity mapping from the input to the output layer. Note that the bottleneck layer acts as a restriction that prevents the trivial solution[1]. In order for an AANN to learn the nonlinear principal components at the output of its bottleneck layer, the number of nonlinear hidden layers must be at least three [1]. However, with a single hidden bottleneck layer, the AANN cannot perform better than singular value decomposition in a reconstruction error sense [6].



$$Cost\ function = \tfrac{1}{2}\left|\left|X - \widehat{X}\right|\right|_2^2 + \lambda \sum_{j=1}^{M} \log\left(1 + \alpha_j^2\right)$$

Figure 1: Block schematic of the three layer sparse auto-associative neural network.

In this paper, instead of restricting the number of nodes to form bottleneck we constrain the hidden layer outputs of an AANN after nonlinearity to be sparse. This is achieved by adding a sparse regularization term to the original reconstruction error cost function and updating the network parameters to minimize the overall cost using the back-propagation algorithm. Fig. 1 shows this basic construction. The resultant network is referred

[1]The trivial solution could arise from each node in a layer being connected with an appropriate nonzero weight to only one node in the subsequent layer such that the network operates in a linear region and further resulting in an identity mapping.

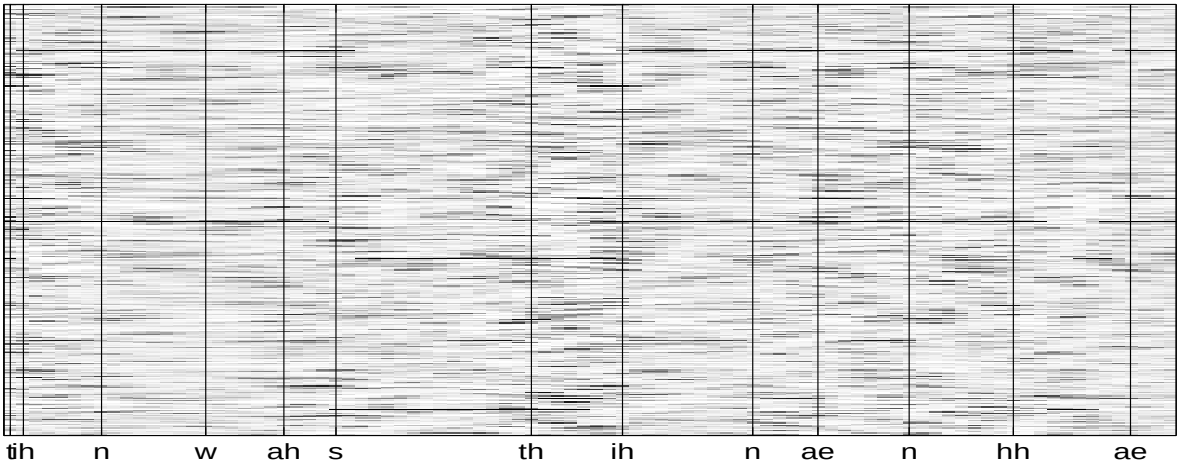


Figure 2: Hidden layer outputs after the nonlinearity of a SAANN corresponding to a sample test utterance. The X-axis marks the beginning of various phonemes and the Y-axis represents the nodes in the hidden layer. SAANN is trained on PLP features and has 350 hidden nodes.

to as the sparse auto-associative neural net (SAANN) throughout this paper.

In the past, hidden layer activations of a multilayer perceptron were successfully used as features for the large vocabulary continuous speech recognition [7],[8]. However both these approaches require a supervised training. The approach proposed here is completely unsupervised as input is mapped to itself.

In a recent work, sparse features were used for acoustic modeling [9]. In that work, features are obtained by first learning a set of basis functions and then expressing a given input as a linear combination of learned basis functions enforcing weights of the linear combination to be sparse. The main drawback with this approach is that a separate linear optimization problem must be solved for every feature vector. The proposed approach in this paper differs in the following ways. First, the transformation from the input to the sparse features is fixed and modeled using part of the SAANN. Secondly, both the transformation and the underlying basis functions[2] are jointly learned using the back-propagation algorithm.

To demonstrate the applicability of the SAANN to speech recognition, we extract features using a single hidden layer SAANN trained using perceptual linear prediction (PLP) cepstral coefficients [10] as both input and output. Once it is trained, the sparse hidden layer outputs after the nonlinearity are used as features for acoustic modeling. Phoneme recognition experiments on TIMIT using the sparse hidden layer features show significant improvement over the baseline PLP features.

The paper is organized as follows. In section 2, we derive the update equations of SAANN. Section 3 describes how SAANN can be used for extracting features for phoneme recognition. Experimental results are presented in section 4. Finally, we conclude in section 5.

2. Sparse Auto-associative Neural Network

Block schematic of the three layer SAANN is shown in the Fig.1. It has N input, N output and M hidden nodes. Input and output layers are linear, whereas the hidden layer is nonlinear with sigmoid nonlinearity. The notations used in this paper are: x_i and y_k denote input to the i^{th} input node and output of the k^{th} output node respectively. The target value at the k^{th} output node is x_k, but is denoted with d_k for clarity i.e., $d_k = x_k$. S_j indicates the accumulated sum before the j^{th} hidden node nonlinearity ϕ, whereas α_j denotes the value after the nonlinearity[3]. The weight between i^{th} input and j^{th} hidden node is w_{ij}, and that of j^{th} hidden and k^{th} output node is c_{jk}. Note that w_{0j} and c_{0k} indicate the bias values at j^{th} hidden and k^{th} output nodes respectively ($x_0 = \alpha_0 = 1$).

The SAANN is trained in an unsupervised manner to minimize the loss function (1) with respect to its parameters ($\{w_{ij}, c_{jk}\}$) over the training data using the mini-batch stochastic gradient descent. The loss function is

[2]For a single hidden layer SAANN, the weights connecting hidden and output units can be interpreted as basis functions.

[3]Specifically, $\alpha_j = \phi(S_j) = \frac{1}{1+e^{-S_j}}$.

given by

$$L = \frac{1}{2}\sum_{k=1}^{N} e_k^2 + \frac{\lambda}{2}\sum_{j=1}^{M} \log\left(1+\alpha_j^2\right) \qquad (1)$$

where $e_k \doteq d_k - y_k$. The first term indicates the squared Euclidean distance between the targets and the outputs of the net. The second term is a sparse regularization [4] which is a function of the outputs of hidden layer after the nonlinearity. Minimizing this regularization term increases the sparsity of the hidden layer outputs α_j. λ is a scalar for controlling the importance of the sparse regularization term relative to the reconstruction error. The parameter update equations to minimize (1) are

$$c_{jk} = c_{jk} - \eta\frac{\partial L}{\partial c_{jk}},\ \forall j \in \{0,1,...,M\},\ \forall k \in \{1,..,N\}$$
$$w_{ij} = w_{ij} - \eta\frac{\partial L}{\partial w_{ij}},\ \forall i \in \{0,1,...,N\},\ \forall j \in \{1,..,M\}$$

where η is a small positive learning rate. The gradient of L with respect to its parameters is computed using the back-propagation algorithm.

2.1. Gradient of L w.r.t. c_{jk}

$$\begin{aligned}\frac{\partial L}{\partial c_{jk}} &= \left(\frac{\partial L}{\partial e_k}\right)\left(\frac{\partial e_k}{\partial y_k}\right)\left(\frac{\partial y_k}{\partial c_{jk}}\right)\\ &= (e_k)(-1)(\alpha_j) = -e_k\alpha_j\end{aligned}$$

2.2. Gradient of L w.r.t. w_{ij}

$$\frac{\partial L}{\partial w_{ij}} = \left\{\frac{\partial L}{\partial \alpha_j}\right\}\left(\frac{\partial \alpha_j}{\partial w_{ij}}\right)$$
$$= \left\{\left(\sum_{k=1}^{N} e_k\,\frac{\partial e_k}{\partial \alpha_j}\right) + \lambda\left(\frac{\alpha_j}{1+\alpha_j^2}\right)\right\}\left(\frac{\partial \alpha_j}{\partial S_j}\,\frac{\partial S_j}{\partial w_{ij}}\right)$$

where,

$$\frac{\partial e_k}{\partial \alpha_j} = \frac{\partial e_k}{\partial y_k}\,\frac{\partial y_k}{\partial \alpha_j} = -c_{jk}$$
$$\frac{\partial \alpha_j}{\partial S_j} = \phi(S_j)(1-\phi(S_j));\ \frac{\partial S_j}{\partial w_{ij}} = x_i$$

Thus,

$$\frac{\partial L}{\partial w_{ij}} = \left\{\left(-\sum_{k=1}^{N} e_k\,c_{jk}\right) + \lambda\left(\frac{\alpha_j}{1+\alpha_j^2}\right)\right\}$$
$$(\phi(S_j)(1-\phi(S_j))\,x_i)$$

The sparse regularization term affects the update of only those weights that are connecting input to the hidden layer, whereas the reconstruction error term affects all parameters.

3. Application to Phoneme Recognition

SAANN described in section 2 is used for extracting features for TIMIT phoneme recognition. The idea is to use the sparse hidden layer outputs after the nonlinearity as features instead of the input representation.

SAANN is trained on PLP features obtained by concatenating a set of 9 frames of standard 13 PLP cepstral coefficients along with its delta and delta-delta features. Thus, the number of input nodes is $39\times 9 = 351$. There is no restriction on the size of the hidden layer since its capacity is controlled by enforcing sparsity. However, 350 hidden nodes are used to keep the dimensionality close to the PLP features. After SAANN is trained, the hidden layer outputs (or SAANN features) are used as input features for the phoneme recognition. SAANN features corresponding to part of a test utterance are shown in Fig. 2.

4. Experiments

Speaker independent phoneme recognition experiments are conducted on the TIMIT database (excluding "sa" dialect sentences) using the hybrid Hidden Markov Model/Multilayer perceptron (HMM/MLP) approach [11]. The training, test and cross-validation (CV) sets of TIMIT consist of 3000, 1344 and 696 utterances from 375, 168 and 87 speakers respectively. All utterances are sampled at 16 kHz. The 61 hand-labeled symbols of the TIMIT transcription are mapped to a standard set of 39 phonemes for the purpose of training and decoding [13].

Initially, a multilayer perceptron (MLP) with a single hidden layer is trained to estimate the posterior probabilities of phonemes conditioned on the input feature vector (either PLP or SAANN features) by minimizing the cross-entropy between its outputs and the corresponding phoneme target classes [12]. In our experiments, MLP has 1000 hidden nodes with sigmoid nonlinearity and 40 output nodes[4] with softmax nonlinearity. The posterior probabilities estimated by the MLP are converted to the scaled likelihoods by dividing them by the corresponding prior probabilities which are then used as the emission likelihoods of the HMM states as described in the hybrid approach [11]. Each phoneme is modeled using 3 HMM states with equal self and transition probabilities. Decoding is accomplished by applying the Viterbi algorithm (no language model) and the phoneme recognition accuracy is obtained by comparing the decoded phoneme sequence against the reference sequence.

Table 1 lists the phoneme recognition accuracies in

[4] Standard 39 phoneme classes along with an additional garbage class.

Table 1: Phoneme recognition accuracies (in %) in clean and noisy (babble) conditions on TIMIT.

Features	clean	20 dB	15 dB	10 dB
PLP (351)	68.2	49.1	38.0	28.7
SAANN (350)	70.0	51.9	40.2	29.7

Table 2: Phoneme recognition accuracy (in %) in clean condition using hierarchical posterior estimation on TIMIT.

Features	clean
PLP (351 features)	70.6
SAANN (350 hidden outputs)	72.1

clean and noisy conditions. Noisy test sets are obtained by adding babble noise (from NOISEX-92) to the clean test set at various signal to noise ratios (SNRs). It can be observed from Table 1 that SAANN features yield a relative improvement in phoneme error rate of 5.6% in clean and 3.2% in noisy conditions respectively, over the baseline PLP features (averaged over all SNRs).

We have also tested the proposed features on a hierarchical phoneme recognition system [14]. The estimates of the posterior probabilities, obtained using a single MLP are further refined by training another MLP. The second MLP, in this case, is trained on a context of 230 ms (i.e., input dimensionality is $40 \times 23 = 920$), and has 1000 hidden and 40 output nodes. Table 2 shows the phoneme recognition accuracies of the hierarchical phoneme recognition system in clean condition. The SAANN features perform significantly better and yield a relative improvement in phoneme error rate of 5.1% over the baseline PLP features.

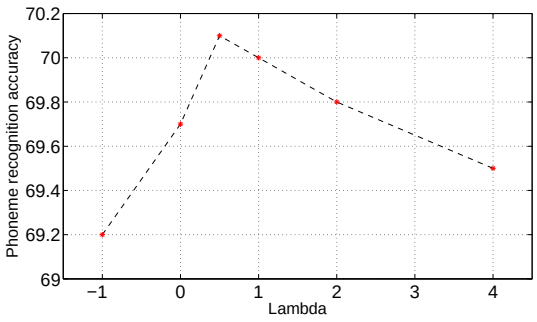


Figure 3: Phoneme recognition accuracy on the test set using SAANN features as a function of Lambda (λ).

In all the above experiments, value of λ is fixed at 1 in the loss function (1). Fig. 3 shows the effect of λ on the phoneme recognition accuracy. For $\lambda = 0$, we obtain a conventional AANN solution. Optimal performance is obtained on this database for $0 < \lambda < 1$, implying effectiveness of the sparsification.

5. Conclusions

In this paper, we proposed sparse auto-associative neural networks and described the modification of the update rule necessary to accommodate the sparse regularization term. Further, we showed its application in unsupervised feature extraction for phoneme recognition. Experiments on state-of-the-art recognition system show a relative improvement in phoneme error rate of 5.1% with the proposed features over the baseline PLP features.

6. References

[1] Kramer, M.A., "Nonlinear principal component analysis using auto-associative neural networks", AIChE Journal, 37(2):233–243, 1991.

[2] Yegnanarayana, B. and Kishore, S.P., "AANN: an alternative to GMM for pattern recognition", Neural Networks, 15(3):459–469, 2002.

[3] Hoyer, P.O., "Non-negative matrix factorization with sparseness constraints", Journal of Machine Learning Research, 5:1457–1469, 2004.

[4] Ranzato, M., Boureau, Y. and LeCun, Y., "Sparse Feature Learning for Deep Belief Networks", Advances in neural information processing systems (NIPS), 20:1185–1192, 2007.

[5] Lee, H. and Ekanadham, C. and Ng, A., "Sparse deep belief net model for visual area V2", Advances in neural information processing systems (NIPS), 2007.

[6] Bourlard, H. and Kamp, Y., "Auto-association by multilayer perceptrons and singular value decomposition", Biological cybernetics, 59(4-5):291–294, 1988.

[7] Chen, B., Zhu, Q. and Morgan, N., "Learning long-term temporal features in LVCSR using neural networks", Eighth International Conference on Spoken Language Processing, 2004.

[8] Grézl, F., Karafiát, M., Kontár, S. and Cernocky, J., "Probabilistic and bottle-neck features for LVCSR of meetings", IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2007.

[9] Sivaram, G.S.V.S., Nemala, S.K, Elhilali, M., Tran, T. and Hermansky, H., "Sparse Coding for Speech Recognition", IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Dallas, Texas, 2010.

[10] Hermansky, H., "Perceptual linear predictive (PLP) analysis of speech", Journal of the Acoustical Society of America, 87(4):1738–1752, 1990.

[11] Bourlard, H. and Morgan, N., "Connectionist speech recognition: a hybrid approach", Neural computation, 1994.

[12] Richard, M.D. and Lippmann, R.P., "Neural network classifiers estimate Bayesian a posteriori probabilities", Neural computation, 3(4):461–483, 1991.

[13] Lee,K.F. and Hon H.W., "Speaker-independent phone recognition using hidden Markov models", IEEE Transactions on Acoustics, Speech and Signal Processing, 37(11):1641–1648, 1989.

[14] Pinto, J., Yegnanarayana, B., Hermansky, H. and Doss, M.M., "Exploiting contextual information for improved phoneme recognition", Proc. of Interspeech, Antwerp, Belgium, 2007.