Binary Coding of Speech Spectrograms Using a Deep Auto-encoder

L. Deng[1], M. Seltzer[1], D. Yu[1], A. Acero[1], A. Mohamed[2], and G. Hinton[2]

[1] Microsoft Research, One Microsoft Way, Redmond, WA 98052, US
[2] University of Toronto, Toronto, Ontario, Canada
{deng|mseltzer|dongyu|alexac}@microsoft.com; {asamir|hinton}@cs.toronto.edu

Abstract

This paper reports our recent exploration of the layer-by-layer learning strategy for training a multi-layer generative model of patches of speech spectrograms. The top layer of the generative model learns binary codes that can be used for efficient compression of speech and could also be used for scalable speech recognition or rapid speech content retrieval. Each layer of the generative model is fully connected to the layer below and the weights on these connections are pre-trained efficiently by using the contrastive divergence approximation to the log likelihood gradient. After layer-by-layer pre-training we "unroll" the generative model to form a deep auto-encoder, whose parameters are then fine-tuned using back-propagation. To reconstruct the full-length speech spectrogram, individual spectrogram segments predicted by their respective binary codes are combined using an overlap-and-add method. Experimental results on speech spectrogram coding demonstrate that the binary codes produce a log-spectral distortion that is approximately 2 dB lower than a sub-band vector quantization technique over the entire frequency range of wide-band speech.

Index Terms: deep learning, speech feature extraction, neural networks, auto-encoder, binary codes, Boltzmann machine

1. Introduction

A recent advance in training methods for multilayer neural networks has led to renewed interest in exploring deep, multi-layer networks for a number of machine learning problems including encoding [8][9], retrieval [12], as well as the problems associated with classification and regression that involves image [8][9], language [16][17] and speech [10][11][15]. The deep networks are first pre-trained, one layer at a time, to form a good generative model of the input data. After pre-training has discovered multiple layers of non-linear features that are good at capturing the structure in the input domain, the whole network is discriminatively fine-tuned. The discriminative fine-tuning can be used to make the network perform good classification or regression, but it can also be used to make the network be good at reconstructing its input from a compact code. While speech researchers have long embraced the concepts of generative modeling and of exploiting the multiple layers of non-linear transformations in the human speech generation process (e.g., [1][2][3][4][5][6]) it was not until the introduction of generative, layer-by-layer pre-training that the full power of generative modeling became apparent. The potential of this approach is illustrated by the strong performance in phonetic recognition, with 78% phonetic recognition accuracy on the TIMIT task, which can be achieved by deep, generative pre-training followed by discriminative fine-tuning [10][11]. Compared with [10][11] which focus on recognition, the current work focuses on the encoding aspect of the new generative modeling paradigm, and it uses log power spectra instead of cepstra to capture greater fine detail of the speech signal.

The work reported in this paper was inspired by the successful use of deep auto-encoders for dimensionality reduction [8][9] and the extension of this work to the discovery of efficient binary codes in information retrieval [12]. It is also motivated by the potential benefits of using discrete representations of speech derived from an almost unlimited supply of unlabeled data in future-generation speech recognition and retrieval systems. In parallel with this work, we are currently also developing discrete speech codes using vector-quantization (VQ) techniques. In particular, we compare coding errors between these two classes of discrete codes.

This paper is organized as follows. In Section 2, we describe the layer-by-layer approach to learning deep generative models introduced in [8]. In section 3, we show how a deep generative model can be used to initialize a deep auto-encoder. We outline the construction of VQ codes as the baseline technique in Section 4. Experimental results showing properties of the codes from the auto-encoder and comparative coding results with the VQ codes are reported in Section 5. Finally, we discuss the implications of results and the potential applications of the discrete speech codes developed from this work in Section 6.

2. Learning a Deep Belief Net

We learn a deep generative model of patches of spectrograms that contain 256 frequency bins and 1, 3, 9, or 13 frames. We first learn an undirected graphical model called a Gaussian-binary restricted Boltzmann machine (RBM) that has one visible layer of linear variables with Gaussian noise and one hidden layer of 500 to 3000 binary latent variables. There is full connectivity between layers, but no connections within either layer. The connection weights (and biases) can be learned efficiently using the contrastive divergence approximation to the log likelihood gradient. For a Gaussian-binary RBM, the conditional density of visible vector, $\boldsymbol{o}$, given a hidden binary vector, $\boldsymbol{h}$, is

$$p(\boldsymbol{o}|\boldsymbol{h}) = \boldsymbol{N}(\boldsymbol{o};\ \boldsymbol{b} + \boldsymbol{h}^T\boldsymbol{W}^T, \boldsymbol{I})$$

and the element-wise conditional distribution of $\boldsymbol{h}$ given $\boldsymbol{o}$ is

$$P(h_j = 1|\boldsymbol{o}) = \sigma(\boldsymbol{c} + \boldsymbol{o}^T\boldsymbol{W})$$

where $\sigma(x) = (1 + e^{-x})^{-1}$. The two conditional distributions can be shown to correspond to the generative model

$$p(\boldsymbol{o}, \boldsymbol{h}) = \frac{1}{Z}\exp(-E(\boldsymbol{o}, \boldsymbol{h}))$$

where $E(\boldsymbol{o}, \boldsymbol{h}) = \frac{1}{2}(\boldsymbol{o} - \boldsymbol{b})^T(\boldsymbol{o} - \boldsymbol{b}) - \boldsymbol{c}^T\boldsymbol{h} - \boldsymbol{o}^T\boldsymbol{W}\boldsymbol{h}$, and Z is the normalization factor, often called the partition function .

After learning the Gaussian-Binary RBM we treat the activation probabilities of its hidden units, when they are being driven by data, as the data for training a binary-binary RBM [8]. These two RBM's can then be composed to form a deep belief net (DBN) in which it is easy to infer the states of the second layer of binary hidden units from the input in a single forward pass [8][7]. The DBN used in this work is illustrated on the left side of Figure 1, where the two RBMs are shown in separate boxes.

The binary states, which are the hidden units of the top RBM, form the code for the input and achieve lower distortion than sub-band VQ (see section 5). Even lower distortion can be achieved by a further step of fine-tuning which we describe now.

3. From Deep Belief Nets to Autoencoders

To fine-tune, we first "unroll" the DBN by using its weight matrices to create a deep, five-layer network whose lower layers use the matrices to encode the input and whose upper layers use the matrices in reverse order to decode the input. This auto-encoder is then fine-tuned using back-propagation of error-derivatives to make its output as similar as possible to its input, as shown on the right side of Figure 1. In our experiments, conjugate gradient is used for the fine-tuning. Details of the training process, including the division of the training set into mini-batches, number of training passes (epochs) in pre-training and fine-tuning, learning rate, momentum, weight decay, and the threshold used to force binary codes, etc., are found to be important to obtain good coding results.

After both pre-training and fine-tuning, we encode and reconstruct any variable-length spectrogram as follows. First, N consecutive overlapping frames of 256-point log power spectra are each normalized to zero-mean and unit-variance to provide the input to the auto-encoder. The first hidden layer then uses the logistic function to compute real-valued activations. These real values are fed to the next, coding layer to compute "codes". The real-valued activations of hidden units in the coding layer are quantized to be either zero or one with 0.5 as the threshold. These binary codes are then used to reconstruct the original spectrogram, where individual fixed-frame patches are reconstructed first using the two upper layers of network weights. Finally, we use the overlap-and-add technique to reconstruct the full- length speech spectrogram from the outputs produced by applying the autoencoder to every possible window of N consecutive frames.

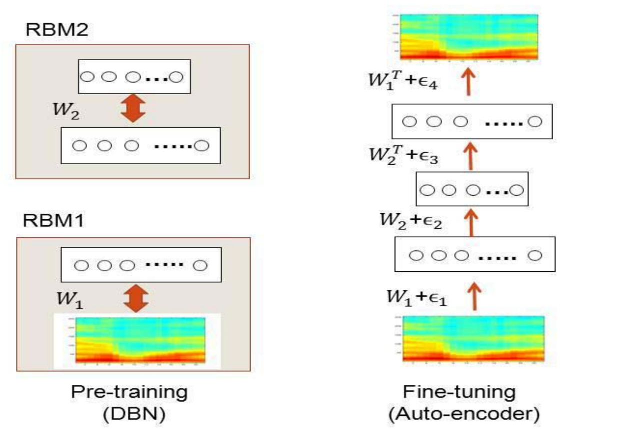

Fig.1. *Left: Illustration of pre-training of the DBN that consists of two RBMs used in this work. Right: Illustration of fine tuning that produces the final deep auto-encoder.*

4. Sub-Band VQ Codes

In order to evaluate the discrete codes generated by the autoencoder described in the previous section, we have also created a method of encoding the N-frames of spectral information using a more traditional approach based on sub-band vector quantization (VQ). Unlike most previous work that uses VQ for speech coding, we are attempting to create a discrete representation for the entire speech spectrum, rather than the smoothed spectrum produced by LPC parameters or MFCC feature vectors. As in the auto-encoder setup, we wish to encode vectors of K*N points, where K is the number of frequency components and N is the number of consecutive frames in a patch. For typical values of K=256 and N=9, performing VQ on such large vectors would be complicated by data sparsity and poor local minima. To prevent this, we divide the spectrum into overlapping spectral bands and quantize the log power spectral coefficients in each sub-band independently. We experimented with both linearly spaced sub-bands and the sub-bands spaced according to the Mel scale, as is common in speech recognition.

In our experiments, the sub-band VQ was performed as follows. Each segment of N consecutive frames of power spectral vectors was divided into 24 patches. The patches represented the spectral content across N frames in each of 24 overlapping frequency ranges. The sub-bands were created using triangular windows similar to the Mel filter-bank. The patches were then windowed in time also using triangular windows. After windowing in time and frequency, the energy in the patch was computed and used to normalize all components in the patch. A two-dimensional DCT was then applied to the logarithm of the patch values and all components except the zero-th coefficient (which is constant for all patches) were retained. The 2D DCT components were then quantized using conventional VQ with the LBG algorithm. The energy components were then scalar quantized. Thus, each of the 24 sub-bands had an associated VQ codebook for the 2D DCT coefficients and a scalar quantization codebook for the energy terms. In our experiments all sub-bands used a codebook size of 2^8 = 256 for the spectral patch information and a codebook size of 2^5 = 32 for the energy values. For 24 sub-bands, this results in a representation in which 24*(8+5) = 312 bits are used to represent the spectral information in a segment of N=9 consecutive frames.

5. Experiments and Results

We have examined properties of the deep autoencoder discussed above, and conducted experiments to compare its coding errors with the better established VQ technique. We would like to capture the fine structures of speech such as harmonics and also the acoustic-phonetic cues that are identifiable in speech spectrograms. We thus moved away from the commonly used Mel-cepstrum coefficients and explored the use of the full power spectrum. The coding errors were found to be unusually large, due partly to the positivity of the data which conflicts with the Gaussian distribution assumed by the linear units in the first and last layers of the autoencoder. We therefore tried using the logarithmic power spectra (spectrogram) and achieved the most encouraging results. We report these results in this section, using 480 male sentences in TIMIT's training set as the training data, and 192 sentences (128 males and 68 females) in TIMIT's core test set as the test data. We also used subsets of male and female speakers, as well as the full mixed set, as our training data in training the auto-encoder, all reaching the same conclusions.

We first qualitatively examine the nature of the codes that the deep auto-encoder uses to represent the real-valued speech data. During the pre-training of the top-level, binary-binary RBM, the code layer is forced to use stochastic binary values for reconstructing its inputs, so it learns to make good use of binary values. During the deterministic fine-tuning, however, the 312 code units (identical bits to the VQ codes) transmit the real-valued probabilities to the next layer instead of sampling. If many of these probabilities are far from 1 and 0, quantizing them to a single bit may cause large distortions. In Fig. 2, we show the histogram, for a typical utterance in TIMIT, of the input speech data (normalized log power spectra) at the top row. This is followed by the histograms of the 312 code-layer units excited by the speech input, with N-frame windows where N=1, 3, 9, and 13, respectively. The units behave in a fairly binary way for reasonably long windows (9 and 13), but not for short windows.

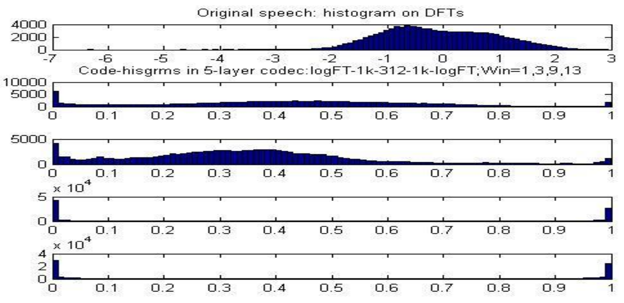

Fig. 2. *Top: the distribution of the input data. Remaining: the distributions of the activations of the 312 code units for input window sizes of 1, 3, 9, and 13, respectively.*

For each of the histograms in Fig. 2, we show the corresponding spectrograms in Fig. 3. At the top is the original speech, followed by the reconstructed speech utterances with forced binary values (zero or one) at the 312 unit code layer for encoding window lengths of N=1, 3, 9, and 13, respectively. The lower coding errors for N=9 and N=13 are clearly seen, which relate to the low quantization errors at the coding layer as demonstrated in Fig. 2.

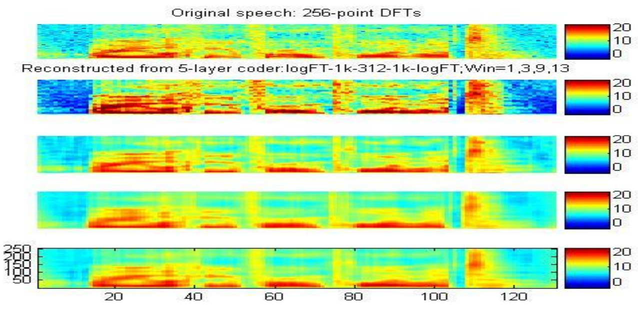

Fig. 3. *Top to Bottom: Original spectrogram; reconstructions using input window sizes of 1, 3, 9, and 13 (forcing the coding units to be zero or one)*

We then qualitatively examine the encoding accuracy of the auto-encoder in comparison with the more traditional VQ codes detailed in Section 4. In Fig.4, we use a typical test utterance to demonstrate the various aspects of encoding accuracy. At the top is the original speech utterance's spectrogram. The next two spectrograms are the blurry reconstruction from the 312-bit VQ and the much sharper reconstruction from the 312-bit auto-encoder. Coding errors from both coders, plotted as a function of time, are shown below the spectrograms, demonstrating that the auto-encoder (red curve) is producing lower errors than the VQ coder (blue curve) throughout the entire span of the utterance. The final two spectrograms show the detailed coding error distributions over both time and frequency bins.

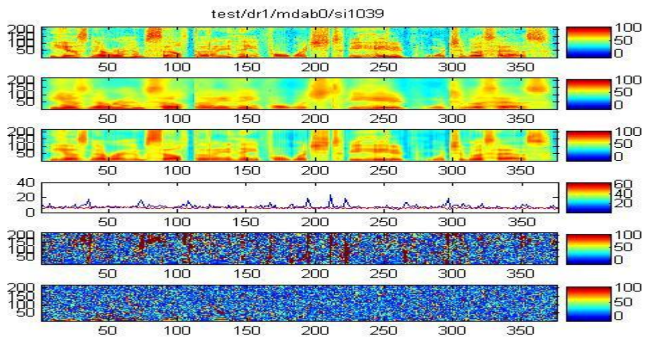

Fig. 4. *Top to bottom: Original spectrogram from the test set; reconstruction from the 312-bit VQ coder; reconstruction from the 312-bit auto-encoder (2304-1000-312); coding errors as a function of time for the VQ coder (blue) and auto-encoder (red); spectrogram of the VQ coder residual; spectrogram of the auto-encoder residual.*

Next, we investigate how the quantitative encoding accuracy of a 312-bit auto-encoder is affected by the number of units in the other two hidden layers and by the use of fine-tuning. The five-layer deep auto-encoders ("Auto" in Table 1) are trained with 480 male utterances in TIMIT's training set and tested on the male, female, and combined sets ("All" in Table 1) of test utterances. "DBN" in table 1 refers to the DBN that was used to initialize the weights of the auto-encoder. Auto-encoders' architectures are denoted by 2304-yyyy-312, where yyyy=500, 1000, 1500, 3000. Specifically, the architecture 2304-1000-312 corresponds to the auto-encoder with a 2304-dimensional input vector (256 frequency bins times 9 frames), 1000 hidden units, and 312 coding units. Note that during fine-tuning, the auto-encoder's architecture becomes 2304-yyyy-312-yyyy-2304.

Ave. Log Spectral Distortion	Males	Females	All
VQ: 312-bits per frame	7.53 dB	7.99 dB	7.68 dB
Auto 2304-500-312	5.57 dB	6.56 dB	5.90 dB
DBN 2304-500-312	6.19 dB	7.01 dB	6.46 dB
Auto 2304-750-312	5.40 dB	6.45 dB	5.73 dB
DBN 2304-750-312	5.93 dB	6.82 dB	6.23 dB
Auto 2304-1000-312	5.42 dB	6.44 dB	5.76 dB
DBN 2304-1000-312	5.82 dB	6.75 dB	6.13 dB
Auto 2304-1500-312	5.69 dB	6.67 dB	6.02 dB
DBN 2304-1500-312	6.11 dB	6.98 dB	6.40 dB
Auto 2304-3000-312	6.46 dB	7.32 dB	6.72 dB
DBN 2304-3000-312	7.22 dB	7.97 dB	7.47 dB

Table 1. *Comparison of coding errors of the VQ technique, the deep auto-encoder with fine-tuning (Auto), and the pre-trained generative model (DBN) that is unrolled to initialize the auto-encoder weights. 312-bit codes are used in all cases. The error measure is mean-square-error of log spectral distortion.*

Examining the encoding errors measured by the standard log spectral distortion (in dB) averaged over all utterances, all frames of each utterance, and over frequency bins, we have the following observations. First, even without fine-tuning as an auto-encoder, all architectures for the pre-trained deep belief net (DBN) that is used to initialize the auto-encoder give lower distortion than the VQ coder using the same number of code bits. Second, fine-tuning as an auto-encoder achieves about twice the reduction in distortion compared with just pre-training the DBN. Third, the auto-encoder's performance depends on the number of the hidden units. This dependency can be a function of the amount of training data and its statistical structure.

The results shown in Table 1 were obtained using a subset of TIMIT training data, due partly to the slow computation with the program written in Matlab run on a CPU. After we acquired a GPU unit, we ran the training on the full TIMIT training set. Similar results were produced. For example, for the 2304-1000-312 auto-encoder shown in Table 1, the error is reduced from 5.76 dB to 5.42 dB. The corresponding VQ coder's error is reduced from 7.68 dB to 7.58 dB. Both are due to the increase of about nine folds in training data.

Finally, we examine in more detail the encoding errors across the frequency range. In the upper graph of Fig. 5, we plot the average log spectral distortions of the VQ and the auto-encoder as a function of frequency. The average is over all frames in all 192 test utterances. Over the entire frequency range, the auto-encoder produces lower errors than the VQ coder. Another measure of coding errors, signal-to-noise ratio or SNR, is also used to compare the two coders, with the comparison results shown in the lower graph of Fig. 5 and with a similar conclusion to the use of log spectral distortion as the error measure.

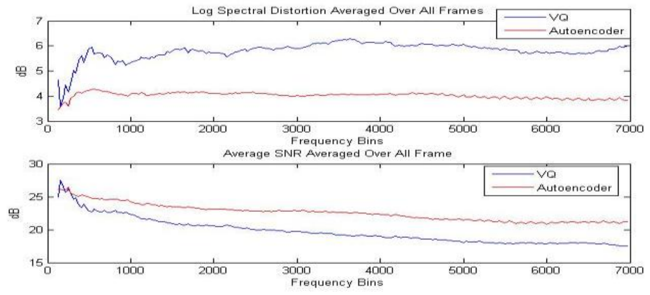

Fig. 5. Comparison of Autoencoder, DBN, and VQ codes using coding error measures of log spectral distortion (upper) and signal-to-noise ratio (lower) as a function of frequency.

6. Summary and Conclusion

The research reported in this paper is a first step towards the automatic discovery of good, discrete representations or "codes" for speech that capture its essential properties for downstream processing such as scalable speech recognition and rapid speech retrieval. We compare the codes found by a more traditional VQ technique with those found by a DBN and by a deep auto-encoder whose parameters are initialized from the DBN using a technique developed originally for image coding and document retrieval [8][12]. It is satisfying to observe that for speech, improvement over a VQ-based coder achieved by using a deep auto-encoder is about the same as previously demonstrated for coding image patches [8].

While strong preliminary results have been obtained, we note that the overlapping property of the input speech data has not been exploited in the current encoding scheme described in this paper. Taking into account how the super vector consisting of N frames is constructed and organized from the raw spectral sequences, we should be able to further improve the encoding effectiveness of the auto-encoder. In addition, our future work aims at the exploitation of the distributed binary codes developed in this work for rapid speech content retrieval and for speech recognition that is scalable over ever-increasing amounts of (unlabeled) training data. To this end, not only do we pay attention to the encoding errors as the focus of this study, but more importantly, we aim to extract essential features that help discriminate different (broad) classes of speech sounds. Some form of discriminative learning commonly used in speech recognition (e.g., [1][7]) need to be modified to adapt to the DBN framework, where an additional class-discriminative term in the objective function is needed for training the auto-encoder with the reconstruction of unlabeled data serving the role of regularization. Also, it is important to condition the discriminative deep auto-encoder's parameters on specific variability factors (e.g., [2][13][14]). These techniques are expected to slightly increase the encoding error but to provide a means to automatically discover powerful and efficient discriminative long-range features to aid downstream applications.

7. References

[1] Baker, J., et. al. "Research developments and directions in speech recognition and understanding," *IEEE Sig. Proc. Mag.,* vol. 26, May 2009, pp. 75-80.

[2] Baker, J., et. al. "Updated MINDS report on speech recognition and understanding," *IEEE Sig. Proc. Mag.*, July 2009, vol. 26, pp. 78-85.

[3] Bell, G., Fujisaki, H., Heinz, J., Stevens, K., and House, A. "Reduction of speech spectra by analysis-by synthesis techniques," J. Acoust. Soc. Am., vol. 33, 1961, pp. 1725-1736.

[4] Deng, L. "Computational Models for Speech Production," chapter in Computational Models of Speech Pattern Processing, pp. 199-213, Springer, 1999.

[5] Deng, L., Yu, D., and Acero, A. "Structured speech modeling," IEEE Trans. Audio, Speech & Language Proc., vol. 14, no. 5, pp. 1492-1504, September 2006.

[6] Halle, M. and Stevens, K. "Speech recognition: A model and program for research,'' IRE Trans. Information Theory, 1962.

[7] He, X., Deng, L., Chou, W. "Discriminative Learning in Sequential Pattern Recognition --- A Unifying Review for Optimization-Oriented Speech Recognition," *IEEE Sig. Proc. Mag.*, vol. 25, 2008, pp. 14-36.

[8] Hinton, G., Osindero, S., and Teh, Y. "A fast learning algorithm for deep belief nets," Neural Computation, vol. 18, pp. 1527-1554, 2006.

[9] Hinton, G. and Salakhutdinov, R. "Reducing the dimensionality of data with neural networks," Science, vol. 313. no. 5786, pp. 504 - 507, July 2006.

[10] Mohamed,A.,Dahl, G.,Hinton, G. "Deep belief networks for phone recognition,"Proc.NIPS Workshop, Dec. 2009.

[11] Mohamed, A., Yu, D., and Deng, L. "Investigation of full-sequence training of deep belief networks for speech recognition," Proc. Interspeech, Sept. 2010.

[12] Salakhutdinov R. and Hinton, G. "Semantic hashing," Proc. SIGIR Workshop on Information Retrieval and Applications of Graphical Models, 2007.

[13] Yu, D., Deng, L., Gong, Y. and Acero, A. "A novel framework and training algorithm for variable-parameter hidden Markov models," *IEEE Trans. Audio, Speech and Language Processing*, vol. 17, 2009, pp. 1348-1360.

[14] Yu, D. and Deng, L. "Solving nonlinear estimation problems using Splines," *IEEE Sig. Proc. Mag.*, vol. 26, 2009, pp. 86-90.

[15] Yu, D. and Deng, L. "Deep-structured hidden conditional random fields for phonetic recognition," Proc. Interspeech, Sept. 2010.

[16] Yu, D., Deng, L., and Wang, S., "Learning in the deep-structured conditional random fields," Proc. NIPS Workshop, Dec. 2009.

[17] Yu, D., Wang, S., Karam, Z., Deng, L. "Language recognition using deep-structured conditional random fields," Proc. ICASSP, April 2010, pp. 5030-5033.