NOISY TRAINING FOR DEEP NEURAL NETWORKS

Xiangtao Meng[1,2], *Chao Liu*[1], *Zhiyong Zhang*[1], *Dong Wang*[1]

1. Center for Speech and Language Technology (CSLT)
Research Institute of Information Technology (RIIT), Tsinghua University
2. School of Compute Science and Technology
Chongqing University of Posts and Telecommunications (CUPT)

ABSTRACT

Deep neural networks (DNN) have gained remarkable success in speech recognition, partially attributed to its flexibility in learning complex patterns of speech signals. This flexibility, however, may lead to serious over-fitting and hence miserable performance degradation in adverse environments such as those with high ambient noises. We propose a noisy training approach to tackle this problem: by injecting noises into the training speech intentionally and randomly, more generalizable DNN models can be learned. This 'noise injection' technique has been well-known to the neural computation community, however there is little knowledge if it would work for the DNN model which involves a highly complex objective function. The experiments presented in this paper confirm that the original assumptions of the noise injection approach largely holds when learning deep structures, and the noisy training may provide substantial performance improvement for DNN-based speech recognition.

Index Terms— deep neural network, noise injection, robust speech recognition

1. INTRODUCTION

Deep neural networks (DNN) gained much attention recently in automatic speech recognition (ASR). Numerous experiments demonstrated that DNN-based systems can achieve much higher recognition accuracy than conventional systems that are based on the Gaussian mixture model (GMM) [1, 2]. Part of the DNN success can be attributed to the large parameter space which provides great flexibility for DNN to learn complex speech patterns from large amounts of data. This flexibility, however, may cause serious over-fitting problems, hence leading to miserable performance reduction in adverse environments such as those with noises. This problem is more evident when the training data is limited and/or channel mismatch between training and testing exists. For example, when the training data is mostly clean and the test data is corrupted by noises, ASR performance usually suffers a substantial reduction [3].

A multitude of research has been conducted to improve noise robustness for DNNs. Multi-condition training was presented in [4], where DNNs were trained by involving speech data in various channel/noise conditions. This approach is straightforward and usually delivers good performance, though collecting multi-condition data is not always possible. Another direction is to use noise-robust features, e.g., auditory feature based on Gammatone filters [3]. Various feature compensation approaches were also studied. [5] proposed to compensate input features using the vector Taylor series (VTS) in an adaptive training framework. The authors of [6] investigated several popular speech enhancement approaches and found that the maximum likelihood spectral amplitude estimator (MLSA) is the best spectral restoration method for DNNs trained with clean speech and tested on noisy data. Some other research involves noise information in the input of the DNN structure and then train 'noise aware' networks. For instance, [7] uses the VTS as a noise estimator to generate noise-dependent inputs for DNNs.

Another promising technique for noise robustness is the denoising auto-encoder (DAE) [8]. The DAE learns a denoising function using an auto-encoder structure which artificially injects noises to the input and reconstructs the original input on the output. Note this approach is not particular for DNNs but a general denoising technique; interestingly, the DAE structure is usually a DNN itself. [9] extended the DAE by introducing recurrent structures and demonstrated that the deep and recurrent auto-encoder provides better performance for ASR in most cases of noises.

In this paper, we propose a noisy training approach for DNNs. The idea is simple: by randomly selecting some noises to corrupt the input speech when conducting DNN training, the noise patterns can be learned, and generalization capability of the resulting network is expected to be improved. Both may improve robustness of DNNs on noisy data.

In the next section we discuss some related works. The main idea of noisy training will be presented in Section 3 and the experimental justification will be presented in Section 4. The entire paper will be concluded by Section 5.

2. RELATED WORK

The noisy training approach proposed in this paper is highly motivated by the noise injection technique which has been studied for a long time in neural computation [10, 11, 12, 13]. This paper extends these studies in two aspects: first, we examine the behavior of noise-injection in DNN training which is a more challenging task; second, we study mixture of multiple noises at various levels of signal-to-noise ratios(SNR), which is different from the conventional noise injection that assumes small and Gaussian-like injected noises.

Another related work of this study is the DAE approach [8, 9]. Both the DAE and noisy training sample random noises to corrupt input signals, though the DAE targets at signal / feature recovery while the noisy training proposed in this paper targets at classification of context-dependent states.

This research was supported by the National Science Foundation of China (NSFC) under the project No. 61371136. It was also supported by the Tencent Corporation and the Huilan Ltd.

Finally this work is also related to the multi-condition training [4] in the sense that the training takes noises of multiple conditions. However, our study focuses on random noise corruption, which on the one hand learns noise patterns as the multi-condition training, and on the other hand, it changes the cost function of the training task so that generalization capability of the resulting DNN is improved.

3. NOISY TRAINING

The basic idea of DNN noisy training is as follows: firstly sample some segments from some real-life noise recordings, and then mix these noise segments with the original training speech; finally employ the corrupted speech data to train the DNN as usual. The rationale of this approach is twofold: firstly the noise patterns within the introduced noise signals can be learned and thus compensated in the test, which is straightforward and similar to the multi-condition training; secondly, the noises provide some perturbation in model training so that generalization capability of the DNN can be improved, which is supported by the noise injection theory. We therefore start from noise injection and then present how to realize the noisy training.

3.1. Noise injection

It has been known for two decades that using noises to corrupt input features in neural network training can improve generalization capability of the resulting network [14]. A bunch of theoretical studies have been presented to understand the implication of this 'noise injection'. Now it is clear that involving a small magnitude of noise in the input is equivalent to introducing a certain regularity in the objective function, which in turn encourages the network converging to a smoother mapping function [15]. More precisely, with noise injection, the training favors an optimal solution at which the objective function is less sensitive to the change of the input [11]. Further studies showed that noise injection is closely correlated to some other well-known techniques, including sigmoid gain scaling and target smoothing by convolution [16], at least with Gaussian noises and shallow networks such as multi-layer perceptrons (MLP) with a single layer. The relationships among regularization, weight decay and noise injection, on the one hand, provide a better understanding for each individual technique, and on the other hand, motivate some novel and efficient algorithms. For example, Bishop showed that noise injection can be approximated by a Tikhonov regularization on the square error cost function [12]. Finally, we note that noise injection can be conducted in different ways, such as perturbation on weights and hidden units [10], though we just consider the noise injection on input units in this paper.

In order to highlight the rationale of noise injection (and so noisy training), we reproduce the formulation and derivation in [11] but migrate the derivation to the case of cross entropy cost which is usually used in classification problems such as ASR.

First of all, formulate an MLP as a nonlinear mapping function $f_\theta : \mathcal{R}^M \longmapsto \mathcal{R}^K$ where M is the input dimension and K is the output dimension, and θ encodes all the parameters of the network including weights and biases. Let $\mathbf{x} \in \mathcal{R}^M$ denotes the input variable, and $\mathbf{y} \in \{0,1\}^K$ denotes the target label which follows the 1-of-K coding scheme. The cross entropy cost is defined as follows:

$$E(\theta) = -\sum_{n=1}^{N}\sum_{k=1}^{K}\{\mathbf{y}^{(n)} ln f_k(\mathbf{x}^{(n)})\}$$

where n indexes the training samples and k indexes the output units. Considering an identical and independent noise $\mathbf{v}$ whose first and second moments satisfy the following constraints:

$$\mathbb{E}\{\mathbf{v}\} = 0 \qquad \mathbb{E}\{\mathbf{v}^2\} = \epsilon I,$$

where I is the M-dimensional identity matrix. Applying the Taylor series of $lnf(\mathbf{x})$, the cost function with the noise injection can be derived as follows:

$$\begin{aligned} E_v(\theta) &= -\sum_{n=1}^{N}\sum_{k=1}^{K}\{\mathbf{y}_k^{(n)} ln f_k(\mathbf{x}^{(n)} + \mathbf{v}^{(n)})\} \\ &\approx -\sum_{n=1}^{N}\sum_{k=1}^{K}\{\mathbf{y}_k^{(n)} ln f_k(\mathbf{x}^{(n)})\} \\ &- \sum_{n=1}^{N}\sum_{k=1}^{K}\mathbf{y}_k^{(n)}\{\mathbf{v}^{(n)^T}\frac{\bigtriangledown f_k(\mathbf{x}^{(n)})}{f_k(\mathbf{x}^{(n)})} + \frac{1}{2}\mathbf{v}^{(n)^T} H_k(\mathbf{x}^{(n)})\mathbf{v}^{(n)}\} \end{aligned}$$

where $H_k(x)$ is defined as follows:

$$H_k(x) = \frac{-1}{f_k(\mathbf{x})} \bigtriangledown f_k(\mathbf{x}) \bigtriangledown f_k(\mathbf{x})^T + \frac{1}{f_k^2(\mathbf{x})} \bigtriangledown \bigtriangledown f_k(\mathbf{x}).$$

Since $\mathbf{v}^{(n)}$ is independent of $\mathbf{x}^{(n)}$ and $\mathbb{E}\{\mathbf{v}\} = 0$, the first order item vanishes and the cost is written as:

$$E_v(\theta) \approx E(\theta) - \frac{\epsilon}{2}\sum_{k=1}^{K} tr(\tilde{H}_k) \tag{1}$$

where

$$\tilde{H}_k = \sum_{n \in \mathcal{C}_k} H_k(\mathbf{x}^{(n)})$$

and $\mathcal{C}_k$ is the set of indices of the training samples belonging to the k-th class.

In order to understand the implication of (1), an auxiliary function can be defined as follows:

$$E(\theta, \mathbf{v}) = -\sum_{n=1}^{N}\sum_{k=1}^{K}\{\mathbf{y}_k^{(n)} ln f_k(\mathbf{x}^{(n)} + \mathbf{v})\}$$

where $\mathbf{v}$ is a small change to the input vectors $\{\mathbf{x}^{(n)}\}$. Note that $E(\theta, \mathbf{v})$ differs from $E_v(\theta)$: $\mathbf{v}$ in $E(\theta, \mathbf{v})$ is a fixed value for all $\mathbf{x}^{(n)}$ while $\mathbf{v}^{(n)}$ in $E_v(\theta)$ is a random variable and differs for each training sample. The Laplacian of $E(\theta, \mathbf{v})$ with respect to $\mathbf{v}$ is computed as follows:

$$\begin{aligned}\bigtriangledown^2 E(\theta,\mathbf{v}) &= tr\{\frac{\partial^2 E(\theta,\mathbf{v})}{\partial \mathbf{v}^2}\} \\ &= -tr\{\sum_{n=1}^{N}\sum_{k=1}^{K}\mathbf{y}_k^{(n)} H_k(\mathbf{x}^{(n)}+\mathbf{v})\} \\ &= -tr\{\sum_{k=1}^{K}\sum_{n\in\mathcal{C}_k} H_k(\mathbf{x}^{(n)}+\mathbf{v})\} \qquad (2)\end{aligned}$$

Comparison of (2) and (1) gives:

$$E_v(\theta) \approx E(\theta) + \frac{\epsilon}{2}\bigtriangledown^2 E(\theta,0). \qquad (3)$$

Equation (3) indicates that injecting noises in the input is equivalent to placing a regularization on the cost function. This regularization is related to the second order derivatives of the cost function with respect to the input, and its strength is controlled by the magnitude of the injected noise. Since $\bigtriangledown^2 E(\theta,0)$ is positive at the optimal solution of θ, the regulated cost function tends to accept solutions with a smaller curvature of the cost. In other words, the new cost function $E_v(\theta)$ is less sensitive to the change on inputs, and therefore may lead to better generalization capability. Note that this result is identical to the one obtained in [11] where the cost function is the square error.

3.2. Noisy deep learning

The derivation in the previous section provides theoretical justification for our noisy training; however, it is still unclear if this kind of training scheme works for the DNN model which involves a large number of parameters and thus tends to exhibit a highly complex cost function. Particularly, the derivation of (3) assumes small noises with diagonal covariances, while in practice we wish to learn complex noise patterns that may be large in volume and fully dimensional correlated. We therefore investigate how the noise injection works for DNN training when the injected noises are in large volume and real-life. In order to simulate noises in practical scenarios, the following scheme is designed.

For each speech signal (utterance), we first choose which type of noise to use to corrupt it. Assuming that there are n types of noises, we randomly select a noise type following a multinomial distribution:

$$v \sim Mult(\mu_1, \mu_2, ..., \mu_n).$$

The parameters $\{\mu_i\}$ are sampled from a Dirichlet distribution:

$$(\mu_1, \mu_2, ..., \mu_n) \sim Dir(\alpha_1, \alpha_2, ..., \alpha_n)$$

where the parameters $\{\alpha_i\}$ are manually set to control the base distribution of the noise type selection. This hierarchical sampling approach (Dirichlet followed by multinomial) simulates various operation environments where the noise type distributions are different. Note that we allow a special noise type: 'no-noise' which means that the speech signal is not corrupted.

Secondly, sample the noise level (i.e., SNR). This sampling follows a Gaussian distribution $\mathcal{N}(\mu_{SNR}, \sigma_{SNR})$ where μ_{SNR} and σ_{SNR} are both manually defined. If the noise type is no-noise, then the SNR sampling is not needed.

The last step is to sample an appropriate noise segment according to the noise type. This is achieved following a uniformed distribution, i.e., randomly select a starting point in the noise speech, and then excerpt a segment of the same length as the speech signal to corrupt.

Finally, the selected noise segment is scaled to reach the required SNR level, and then is mixed with the original speech signal. The noise-mixed speech is fed into the DNN to conduct network training.

4. EXPERIMENTS

4.1. Databases

The experiments were conducted with the wall street journal database. The setting is largely standard: the training part used the wsj si284 training dataset, which involves 37318 utterances or about 80 hours of speech signals. The wsj dev93 dataset (503 utterances) was used as the development set for parameter tuning and cross validation in DNN training. The wsj eval93 dataset (213 utterances) was used to conduct evaluation.

Note that the wsj database was recorded in a noise-free condition. In order to simulate noise-corrupted speech signals, the DEMAND noise database[1] was used to sample noise segments. This database involves 18 types of noises, from which we selected 2 types (white and cafeteria) as 'known noise' and 6 types (car, restaurant, train station, bus, park) as 'unknown noise'. The known noises are used to verify the capability of noise pattern learning of the proposed approach, while the unknown noises are used to test its generalizability.

4.2. Experimental settings

We used the Kaldi toolkit[2] to conduct the training and evaluation, and largely followed the wsj s5 recipe. Specifically, the standard MFCC is used as the feature to train a GMM system which involves 351 phones and 3447 Gaussian mixtures. The DNN system is then trained utilizing the alignments provided by the GMM system, based on the 40-dimensional Fbank feature. A symmetric 11-frame window is applied to concatenate neighboring frames, and an LDA transform is used to reduce the feature dimension to 200, which is used as the DNN input feature.

The DNN architecture involves 4 hidden layers and each layer consists of 1200 units. The output layer is composed of 3447 units, equal to the total number of Gaussian mixtures in the GMM system. The cross entropy is set as the objective of DNN training, and the stochastic gradient descendent (SGD) approach is employed to perform optimization, with the mini batch size set to 256 frames. This setting is quite close to the GPU recipe used in Kaldi.

In order to inject noises, the signal energy is computed for each training/test utterance, and a noise segment is randomly selected and scaled according to the expected SNR; the speech and noise signals are then mixed by simply time-domain addition. Note that the noise injection is conducted before an utterance-based cepstral mean normalization (CMN). In noisy training, the training data is corrupted by noise, while the cross validation data remains clean. Note that the process of the model training is reproducible in spite

[1] http://parole.loria.fr/DEMAND/

[2] http://kaldi.sourceforge.net/

of the randomness of the noise injection, since the random seed is hard-coded.

In test, the noise type and SNR are all fixed so that we can evaluate system performance in a specific noise condition. This is different from the training phase where both the noise type and SNR can be random. We choose the 'big dict' test case suggested in the Kaldi wsj recipe, which is based on a large dictionary consisting of 150k English words and a corresponding 3-gram language model.

4.3. Known noise injection

We first study injecting the same noise in training and test, which can probe the capability of the proposed method in noise pattern learning. We choose the two 'known noises' to conduct the experiment: white noise and cafeteria noise, and study two different injection methods: the narrow band injection where the SNR variance is set close to 0 (σ_{SNR} = 0.01) so the noise level concentrates at a particular value defined by μ_{SNR}, and the broad band injection where the SNR variance is set to a large value ($\sigma_{SNR} = 10$), which allows substantially changed noise magnitudes. Note that the noises injected to the test utterances are always narrow band.

The recognition performance in terms of word error rates (WER) are presented in Fig. 1 and Fig. 2 for white noise injection and cafeteria noise injection respectively. In the figures, the narrow band injection is presented as solid lines and the broad band injection is presented as bars. We can observe that the baseline system trained with clean speech (the blue line) is highly vulnerable against noise corruption, particularly in the case of white noise. With noisy training, the performance on noise conditions is remarkably improved. We also find that the narrow band injection is the most effective on test conditions with matched SNRs (which can be regarded as the upper bound of performance of the multi-condition training). This is expected as the noisy training teaches noise patterns at a particular SNR. In contrast, the broad band noise injection is generally effective: in almost all the cases, injecting broad band noise with $\mu_{SNR} = 15$db performs the best, and more interestingly, this noise injection does not substantially reduce performance on clean speech. This is highly promising because it enables us to treat noises with different SNRs with a single training.

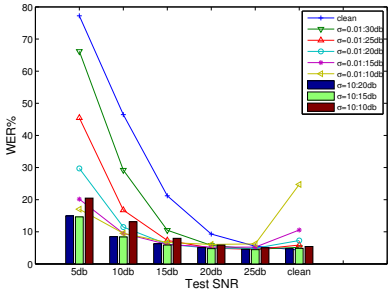

Fig. 1. *WER with white noise injection.*

4.4. Unknown noise injection

In the second experiment, we examine the noisy training for unknown noises. Specifically, we use the known noises (white noise and cafeteria noise) to train the DNN model, and then test it against speech signals corrupted by the 6 types of unknown noises. It deserves to highlight that the special 'no-noise' type is also involved in the Dirichlet sampling (ref. Section 3.2), and distributed uniformly together with the other two noise types. The SNR sampling concentrates at 15db with a variance of 10, which is the best configuration obtained in the previous experiment. The WER results are shown in Fig. 3, where the lines present performance of the baseline system, and the bars present performance with noisy training. We see that in almost all the conditions, noisy training provides significant performance improvement, demonstrating its effectiveness in learning true patterns of human speech. There is an exception, however, in the case of car noise, where the performance with noisy training is almost the same as that of the baseline. This is probably due to the fact that the car noise is significantly different from the two types of noises involved in training.

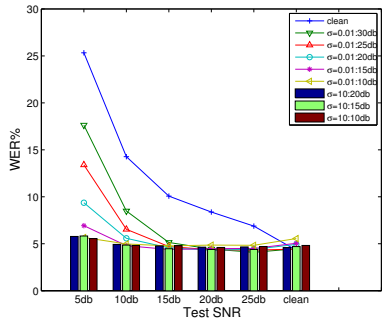

Fig. 2. *WER with cafeteria noise injection.*

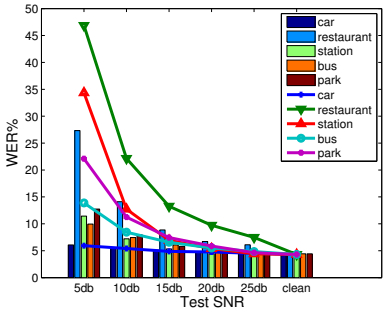

Fig. 3. *WER with unknown noise injection. The model was trained with white noise and cafeteria noise injected.*

5. CONCLUSIONS

We proposed a noisy training approach for DNN-based speech recognition. The analysis and experiments showed that by injecting some noises in input speech, the noise patterns can be effectively learned and the generalization capability of the learned DNNs can be improved. Both the two advantages result in substantial performance improvement with DNN-based ASR systems in noise conditions. Future work involves investigating various noise injection approaches (e.g., weighted noise injection) and evaluating with more noise types.

6. REFERENCES

[1] George E Dahl, Dong Yu, Li Deng, and Alex Acero, "Large vocabulary continuous speech recognition with context-dependent DBN-HMMs," in *Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on*. IEEE, 2011, pp. 4688–4691.

[2] Geoffrey Hinton, Li Deng, Dong Yu, George E Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara N Sainath, et al., "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *Signal Processing Magazine, IEEE*, vol. 29, no. 6, pp. 82–97, 2012.

[3] Jun Qi, Dong Wang, Ji Xu, and Javier Tejedor, "Bottleneck features based on gammatone frequency cepstral coefficients," in *INTERSPEECH*, 2013.

[4] Dong Yu, Michael L Seltzer, Jinyu Li, Jui-Ting Huang, and Frank Seide, "Feature learning in deep neural networks - a study on speech recognition tasks," in *Proc. International Conference on Learning Representations*, 2013.

[5] Bo Li and Khe Chai Sim, "Noise adaptive front-end normalization based on vector taylor series for deep neural networks in robust speech recognition," in *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*. IEEE, 2013, pp. 7408–7412.

[6] Bo Li, Yu Tsao, and Khe Chai Sim, "An investigation of spectral restoration algorithms for deep neural networks based noise robust speech recognition," in *INTERSPEECH*, 2013.

[7] Michael L. Seltzer, Dong Yu, and Yongqiang Wang, "An investigation of deep neural networks for noise robust speech recognition," in *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*. IEEE, 2013.

[8] Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol, "Extracting and composing robust features with denoising autoencoders," in *Proceedings of the 25th international conference on Machine learning*. ACM, 2008, pp. 1096–1103.

[9] Andrew L Maas, Quoc V Le, Tyler M O'Neil, Oriol Vinyals, Patrick Nguyen, and Andrew Y Ng, "Recurrent neural networks for noise reduction in robust ASR," in *INTERSPEECH*, 2012.

[10] Guozhong An, "The effects of adding noise during backpropagation training on a generalization performance," *Neural Computation*, vol. 8, no. 3, pp. 643–674, 1996.

[11] Yves Grandvalet and Stéphane Canu, "Comments on "noise injection into inputs in back propagation learning"," *Systems, Man and Cybernetics, IEEE Transactions on*, vol. 25, no. 4, pp. 678–681, 1995.

[12] Chris M Bishop, "Training with noise is equivalent to tikhonov regularization," *Neural computation*, vol. 7, no. 1, pp. 108–116, 1995.

[13] Yves Grandvalet, Stéphane Canu, and Stéphane Boucheron, "Noise injection: Theoretical prospects," *Neural Computation*, vol. 9, no. 5, pp. 1093–1108, 1997.

[14] Jocelyn Sietsma and Robert JF Dow, "Neural net pruning-why and how," in *Neural Networks, 1988., IEEE International Conference on*. IEEE, 1988, pp. 325–333.

[15] Kiyotoshi Matsuoka, "Noise injection into inputs in back-propagation learning," *Systems, Man and Cybernetics, IEEE Transactions on*, vol. 22, no. 3, pp. 436–440, 1992.

[16] Russell Reed, RJ Marks, Seho Oh, et al., "Similarities of error regularization, sigmoid gain scaling, target smoothing, and training with jitter," *Neural Networks, IEEE Transactions on*, vol. 6, no. 3, pp. 529–538, 1995.