Evolution Strategy Based Neural Network Optimization and LSTM Language Model for Robust Speech Recognition

Tomohiro Tanaka[1], *Takahiro Shinozaki*[1],
Shinji Watanabe[2], *Takaaki Hori*[2]

[1]Tokyo Institute of Technology, Japan
[2]Mitsubishi Electric Research Laboratories, USA

Abstract

This paper reports our system for the 1-channel track task in the 4th CHiME challenge (CHiME4). A bottle-neck in developing neural network based systems is the tuning of meta-parameters. We automate it by using Covariance Matrix Adaptation Evolution Strategy (CMA-ES) so that high performance system is obtained without relying on human experts. We run two evolution experiments for the DNN acoustic model used in the official baseline system. One uses development set word error rate (WER) after the cross-entropy (CE) based training as the objective function for the evolution, and the other uses the WER after the sequential discriminative training. Additionally, we run an evolution experiment for a Long Short-Term Memory recurrent neural network based language model (LSTM-LM), replacing the original recurrent neural network language model (RNN-LM) used in the baseline system for N-best rescoring. All of these evolution experiments resulted in reduced WERs. To produce the final results, we augmented training data by pooling speech data from all the 6 channels and imported the optimized meta-parameter settings without modification. For the real test data, reduced WER of 17.40% and 16.58% were obtained compared to the baseline WER of 22.75% when the RNN and LSTM-LMs were used, respectively.

1. Background

Neural network based techniques have shown great performance in automatic speech recognition (ASR) tasks [1, 2]. To use neural network, various meta-parameters must be specified including model topology (e.g., the numbers of layers and hidden units), training configuration (e.g., the learning rate and the maximum number of iterations) and system organization (e.g., the choice of features). Properly tuning these meta-parameters is essential for building high performance systems. Usually, the tuning is manually performed. However, it requires expert knowledge and laborious effort. Thus there is a demand to automate the tuning process using computers.

We have previously investigated several automatic meta-parameter optimization frameworks for neural network acoustic models [3, 4, 5]. In the experiments, covariance matrix adaptation-evolution strategy (CMA-ES) [6, 7, 8] showed superior performance than Genetic Algorithm (GA) and Bayesian optimization [9, 10] giving better model with smaller or similar number of system evaluations. Further, we have applied CMA-ES to optimize neural network based language models and have shown that it works well to improve system performance [11]. Here, we apply it to neural network based acoustic and language models in the CHiME4 1-channel track task.

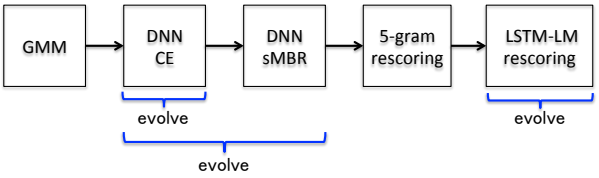

Figure 1: *Recognition system used for evolution of DNN-AM and LSTM-LM.*

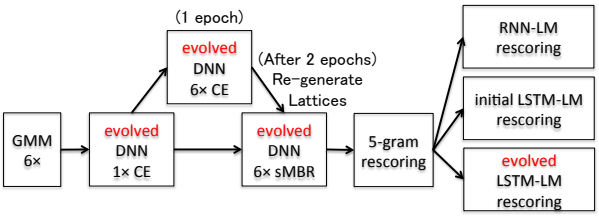

Figure 2: *Recognition system used with augmented acoustic model training data.*

2. Contributions

2.1. CMA-ES based tuning of neural networks

CMA-ES is a population based algorithm for black box optimization that has demonstrated superior performance in several benchmarking tasks. Similar to the GA, it encodes possible solutions as genes. It assumes that the value of an objective function $f(\boldsymbol{x})$ is available, while the functional form of f might be too complex to perform analytical optimization. More specifically, CMA-ES estimates parameters $\boldsymbol{\theta}$ of a Gaussian distribution for a gene $\boldsymbol{x}$ such that the distribution is concentrated in a region with high values of $f(\boldsymbol{x})$ as shown in Eq. (1).

$$\hat{\boldsymbol{x}} \sim \mathcal{N}(\boldsymbol{x}|\hat{\boldsymbol{\theta}}) \ \text{ s.t. } \ \hat{\boldsymbol{\theta}} = \arg\max_{\boldsymbol{\theta}} \underbrace{\int f(\boldsymbol{x})\mathcal{N}(\boldsymbol{x}|\boldsymbol{\theta})d\boldsymbol{x}}_{\triangleq \mathbb{E}[f(\boldsymbol{x})|\boldsymbol{\theta}]} . \quad (1)$$

The estimation of $\boldsymbol{\theta}$ is based on an iterative method, where in each iteration, a set of genes $\{\boldsymbol{x}\}$ is sampled from the Gaussian, their performance $f(\boldsymbol{x})$ is evaluated, and $\boldsymbol{\theta}$ is updated based on the results. In other words, while GA represents a distribution of genes in a generation by the samples themselves, CMA-ES uses a Gaussian distribution. In our case, a gene represents a set of meta-parameters of a neural network to optimize.

Table 1: WER after CE based DNN-AM training.

System	Dev		Test	
	real	simu	real	simu
Baseline	16.45	17.81	29.67	26.20
Evolved	15.40	16.88	29.16	25.28

Table 2: WER after sequential DNN-AM training.

System	Dev		Test	
	real	simu	real	simu
Baseline	14.90	15.70	27.24	24.34
Evolved	13.82	15.49	25.67	22.95

Table 3: WER after RNN/LSTM-LM based rescoring.

System	Dev		Test	
	real	simu	real	simu
Baseline RNN-LM	11.60	12.92	22.75	21.07
+ Evolved DNN-AM	10.98	12.74	21.29	19.74
Evolved LSTM-LM	10.20	12.23	21.09	19.66
+ Evolved DNN-AM	10.00	11.45	20.54	18.85

2.2. LSTM based language model

Neural network based language models have shown to be very effective for improving speech recognition performance [12]. In the CHiME4 baseline system [13], recurrent neural network language model (RNN-LM) [14] is used for final rescoring. The parameters of a RNN are trained using back-propagation through time (BPTT) so that the context dependency is modeled. However, RNNs cannot effectively use long context information due to the vanishing gradient problem [15]. To address the problem, Long Short-Term Memory RNN that utilizes LSTM blocks has been proposed [16]. A LSTM block has a memory cell and three gates (input, forget and output) to control the value stored in the memory cell. By replacing the unit in recurrent hidden layer of a RNN language model with the LSTM block, a LSTM RNN language model (LSTM-LM) [17] is obtained. We replace RNN-LM with LSTM-LM, which is known to perform better in various tasks [18].

3. Experimental evaluation

3.1. Evolution using single channel training data

Using the single channel (channel 5) multicondition training data, we ran two evolution experiments for the DNN acoustic model (DNN-AM) used in the official baseline system based on CMA-ES. One used development set WER after the CE based training as the objective function for the evolution, and the other used the WER after the sequential discriminative training based on state-level Minimum Bayes Risk (sMBR) criterion. Additionally, we ran an evolution experiment for a LSTM-LM replacing the original RNN-LM using WER after N-best rescoring as the objective for the evolution, where 100-best was generated from the decoding result using the 5-gram language model with Kneser-Ney smoothing [19]. These three evolutions were performed independently. Figure 1 shows where the evolutions were performed in the recognition system structure.

For the DNN-AM, 11 meta-parameters were optimized, which were the same as our previous work [5]. These included the number of hidden layers, the number of units per a hidden layer, the initial learning rate, and so on. The population size (e.g. the number of sampled genes from the Gaussian at each generation) was 36. The numbers of iterations (e.g. generations) were 6 and 4 for the two evolutions, respectively. Table 1 shows the results when WER after CE based DNN-AM training was used as the objective, and Table 2 shows the results when WER after the sequential training was used. In both cases, lower WERs were obtained by the evolution based automatic tuning. The sequential training gave some gain compared to the CE based training, and evolution based optimization gave further gain. The best performing DNN chosen by the development set WER had 9 hidden layers and 2461 units per a layer.

For the LSTM-LM, 19 meta-parameters were optimized including the vocabulary size, the number of layers, the number of units per a layer, the initial learning rate and the dropout ratio [20]. The maximum number of hidden layers were set to six and they were used depending on the number of hidden layer. The population size was 30 and the number of generations was 4. All LSTM-LMs were trained using the Chainer toolkit [1] [21]. The population sizes were decided based on our previous experiments and available computer resources for this experiment. Table 3 shows the results. By using LSTM-LM, lower WERs were obtained than the baseline RNN-LM. When the DNN-AM evolved by using the WER after the sequential training was combined, further reduction in WERs was obtained. The vocabulary size of the tuned LSTM-LM was 8112 and the number of hidden layers was 2. The list of the meta-parameters and their initial and optimized values are shown in appendix.

Table 4: WER after RNN/LSTM-LM based rescoring. DNN-AM was trained using the augmented training data.

System	Dev		Test	
	real	simu	real	simu
RNN-LM	9.09	10.86	17.40	16.49
Initial LSTM-LM	9.02	10.82	17.52	16.62
Evolved LSTM-LM	8.06	10.15	16.58	15.67

Table 5: Detailed WERs after RNN-LM rescoring. DNN-AM was trained using the augmented training data.

Env.	Dev		Test	
	real	simu	real	simu
BUS	12.27	9.63	26.51	12.27
CAF	9.23	14.69	19.18	19.11
PED	5.58	8.30	13.62	16.51
STR	9.28	10.83	10.29	18.06
AVG.	9.09	10.86	17.40	16.49

3.2. Single channel system with augmented training data

In the official single channel CHiME4 baseline system, only 5th channel data is used for training. We augmented the training data by 6 times by pooling speech data from all the 6 channels of the official data for further improvement. For the DNN-AM and LSTM-LM, the previously optimized meta-parameters by the evolutions using the original (1x) training data were imported and used as it is. To save the time for experiment, part of the CE based DNN training used the 1x data, and lattice regeneration was performed at slightly different timing from the baseline system as shown in Figure 2. The sequential training for DNN-AM was performed for 6 epochs. Table 4 shows

[1] http://chainer.org/

Table 6: Detailed WERs after LSTM-LM rescoring. LSTM-LM was trained importing the evolved meta-parameters. DNN-AM was trained using the augmented training data.

Env.	Dev		Test	
	real	simu	real	simu
BUS	10.93	9.22	26.00	11.39
CAF	8.29	13.86	18.58	18.77
PED	4.69	7.80	12.05	15.50
STR	8.32	9.73	9.68	17.02
AVG.	8.06	10.15	16.58	15.67

summary of WERs after the RNN-LM based rescoring and the LSTM-LM based rescoring using the initial and the evolved meta-parameters. As can be seen, the lowest WERs were obtained when the evolved LSTM-LM was used. Tables 5 and 6 show the details of the WERs when the RNN and the evolved LSTM-LM were used. By using the LSTM-LM, the averaged real environment WER for the development and evaluation sets were 8.06% and 16.58%, respectively.

4. Acknowledgments

The work of T. Tanaka and T. Shinozaki was supported by JSPS KAKENHI Grant Number 26280055.

5. References

[1] H. Schwenk and J.-L. Gauvain, "Connectionist language modeling for large vocabulary continuous speech recognition." in *Proc. ICASSP*, 2002, pp. 765–768.

[2] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, and B. Kingsbury, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 82–97, 2012.

[3] S. Watanabe and J. Le Roux, "Black box optimization for automatic speech recognition," in *Proc. ICASSP*. IEEE, 2014, pp. 3256–3260.

[4] T. Shinozaki and S. Watanabe, "Structure discovery of deep neural network based on evolutionary algorithms," in *Proc. ICASSP*, 2015, pp. 4979–4983.

[5] T. Moriya, T. Tanaka, T. Shinozaki, S. Watanabe, and K. Duh, "Automation of system building for state-of-the-art large vocabulary speech recognition using evolution strategy," in *Proc. ASRU*, 2015, pp. 610–616.

[6] N. Hansen, S. D. Müller, and P. Koumoutsakos, "Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (CMA-ES)," *Evolutionary Computation*, vol. 11, no. 1, pp. 1–18, 2003.

[7] Y. Akimoto, Y. Nagata, I. Ono, and S. Kobayashi, "Bidirectional relation between CMA evolution strategies and natural evolution strategies," in *Proc. Parallel Problem Solving from Nature (PPSN)*, 2010, pp. 154–163.

[8] D. Wierstra, T. Schaul, T. Glasmachers, Y. Sun, J. Peters, and J. Schmidhuber, "Natural evolution strategies," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 949–980, 2014.

[9] J. Mockus, "On Bayesian methods for seeking the extremum," in *Proceedings of the IFIP Technical Conference*. London, UK, UK: Springer-Verlag, 1974, pp. 400–404.

[10] J. Snoek, H. Larochelle, and R. P. Adams, "Practical Bayesian optimization of machine learning algorithms," in *Advances in Neural Information Processing Systems 25*, 2012.

[11] T. Tanaka, T.Moriya, T. Shinozaki, S. Watanabe, T. Hori, and K. Duh, "Automated structure discovery and parameter tuning of neural network language model based on evolution strategy," in *IEEE Workshop on Spoken Language Technology (SLT)*, 2016, (accepted).

[12] Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin, "A neural probabilistic language model," *Journal of Machine Learning Reseach*, vol. 3, pp. 1137–1155, 2003.

[13] E. Vincent, S. Watanabe, A. A. Nugraha, J. Barker, and R. Marxer, "An analysis of environment, microphone and data simulation mismatches in robust speech recognition," *Computer Speech and Language*, 2016, (submitted).

[14] T. Mikolov, M. Karafiát, L. Burget, J. Cernocký, and S. Khudanpur, "Recurrent neural network based language model," in *Proc. INTERSPEECH*, 2010, pp. 1045–1048.

[15] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE Transactions on Neural Networks*, vol. 5, no. 2, pp. 157–166, 1994.

[16] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[17] M. Sundermeyer, R. Schlüter, and H. Ney, "LSTM neural networks for language modeling," in *Proc. INTERSPEECH*, 2012, pp. 194–197.

[18] T. Hori, C. Hori, S. Watanabe, and J. R. Hershey, "Minimum word error training of long short-term memory recurrent neural network language models for speech recognition," in *Proc. ICASSP*, 2016, pp. 5990–5994.

[19] R. Kneser and H. Ney, "Improved backing-off for m-gram language modeling," *Proc. ICASSP*, pp. 181–184, 1995.

[20] G. E. Dahl, T. N. Sainath, and G. E. Hinton, "Improving deep neural networks for LVCSR using rectified linear units and dropout," in *Proc. ICASSP*. IEEE, 2013, pp. 8609–8613.

[21] S. Tokui, K. Oono, S. Hido, and J. Clayton, "Chainer: a next-generation open source framework for deep learning," in *Neural Information Processing Systems (NIPS)*, 2015.

A. Appendix: Meta-parameters

Table 7 shows the initial and optimized meta-parameters for the DNN acoustic model using the development set WER after the sequential discriminative training as the objective for evolution. Similarly, table 8 lists the initial and optimized meta-parameters for the LSTM language model. For each table, the best gene of the meta-parameters was selected from the pool of all the generations based on the WER of the development set.

Table 7: Meta-parameters for DNN-AM.

Description	Initial value	Best value
feature type({MFCC,FBANK,PLP})	FBANK	FBANK
splice (segment length for DNN	5	7
# of hidden layers	6	9
# of hidden layer units	2048	2461
initial parameters in 1st RBM	$1.00E-1$	$1.15E-1$
initial parameters in other RBMs	$1.00E-1$	$5.04E-2$
RBM learning rate	$4.00E-1$	$5.64E-1$
lower RBM learning rate	$1.00E-2$	$1.26E-2$
RBM Lasso regularization	$2.00E-4$	$1.61E-4$
learning rate for fine tuning	$8.00E-3$	$3.38E-4$
momentum for fine tuning	$1.00E-5$	$9.33E-6$

Table 8: Meta-parameters for LSTM-LM.

Description	Initial value	Best value
vocabulary size	5000	8112
# of hidden layers	2	2
# of projection layer units	300	399
# of 1st layer units	300	671
# of 2nd layer units	300	438
NNLM weight	0.50	0.52
acoustic weight	14.00	21.56
minibatch size	32	35
dropout ratio	0.50	0.44
initial learn rate	1	0.90
learn decay	0.50	0.48
learn decay epochs	6	7
momentum	$1.00E-10$	$1.03E-10$
gradient clipping	5.00	6.23
initial forget gate bias	1.00	1.18