Progressive Neural Network-based Knowledge Transfer in Acoustic Models

Takafumi Moriya*, Ryo Masumura*, Taichi Asami†, Yusuke Shinohara*,
Marc Delcroix‡, Yoshikazu Yamaguchi*, and Yushi Aono*
* NTT Media Intelligence Laboratories, NTT Corporation, Japan
E-mail: moriya.takafumi@lab.ntt.co.jp
† NTT DOCOMO, INC., Japan
‡ NTT Communication Science Laboratories, NTT Corporation, Japan

***Abstract*—This paper presents a novel deep neural network architecture for transfer learning in acoustic models. A well-known approach for transfer leaning is using target domain data to fine-tune a pre-trained model with source model. The model is trained so as to raise its performance in the target domain. However, this approach may not fully utilize the knowledge of the pre-trained model because the pre-trained knowledge is forgotten when the target domain is updated. To solve this problem, we propose a new architecture based on progressive neural networks (PNN) that can transfer knowledge; it does not forget and can well utilize pre-trained knowledge. In addition, we introduce an enhanced PNN that uses feature augmentation to better leverage pre-trained knowledge. The proposed architecture is challenged in experiments on three different recorded Japanese speech recognition tasks (one source and two target domain tasks). In a comparison with various transfer learning approaches, our proposal achieves the lowest error rate in the target tasks.**

I. Introduction

Using deep neural network (DNN) based acoustic models [1] in automatic speech recognition (ASR) systems demands training data for various domains as recently popular voice assistant products will be used in various conditions. Developers collect training data of a new domain so as to improve system performance in that domain. The data is combined with all data in hand and used to re-train an existing multi-condition model. However, developers face two problems as shown in Fig. 1. Some data can be lost due to time limits for use (data containing personal information often has relatively short storage life), and the model cannot be re-trained with full training data when using approaches designed for multi-condition training [2], [3] and multi-task learning [4]–[6]. Additionally, these approaches incur huge time costs if re-training attempts to utilize the full set of training data. Given this background, we focus on a transfer learning approach that leverages a model trained with source domain data to achieve enhanced target domain performance.

Transfer learning [4]–[13] can utilize pre-trained knowledge and so improves the performance in the target domain compared to models trained using only target domain data. A well-known approach for transfer leaning that uses only target domain data is fine-tuning of a pre-trained model [7], [8], called PT/FT. While PT/FT can raise ASR performance in the target domain, it drops the knowledge accumulated in the pre-trained model. We assume that the pre-trained model should extract fixed features to preserve the accumulated knowledge. Knowledge distillation is a domain adaptation approach [10]–[13] that can be also used for transfer learning. This approach distills useful knowledge from the pre-trained model for inclusion in the target domain model. It works well if the pre-trained model contains a large amount of knowledge. We need a framework that allows even models trained with a limited amount of data to achieve good performance. Moreover the approaches of PT/FT and knowledge distillation do not explicitly preserve the pre-trained knowledge in the target domain model.

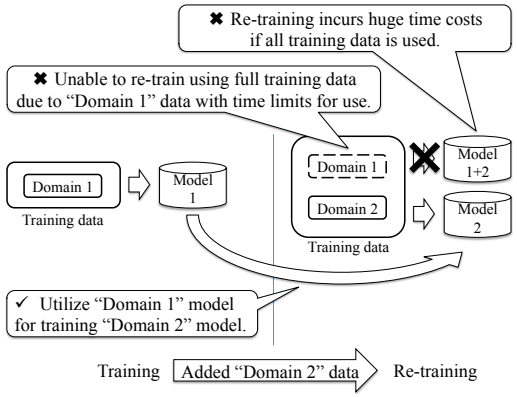

Fig. 1. *Facing real problems in re-building a model. When "Domain 2" data is added, some data may be lost due to usage time limits (data containing personal information often has relatively short storage life) such as "domain 1" data. Developers cannot utilize all training data if re-training approaches designed for multi-condition training and multi-task learning are used. We want to effectively utilize the pre-trained knowledge of "Domain 1" for training "Domain 2" model.*

In order to perform transfer learning while keeping the pre-trained knowledge, we focus on progressive neural networks (PNN) [14]–[17] as they offer continual learning [14], [18], [19], ensemble learning [20]–[22] and transfer learning simultaneously. PNN can be trained using just target domain data without forgetting the pre-trained knowledge; the target domain model is composed of a newly added model and a frozen pre-trained model. In [16], PNN was applied to emotion recognition and shown to yield better results than the PT/FT approach. The results were very promising and indicated that PNN can effectively transfer the pre-trained knowledge to

the target domain task. In [17], PNN was applied to an acoustic model and attained the best performance compared with multi-conditioned- and dependent-conditioned-models. Unfortunately, the performance improvement was slight because PNN cannot fully leverage the pre-trained knowledge. We assume that PNN should also receive explicitly augmented input feature that reflects the source domain. In this paper, we propose a new PNN-based archictecture that can utilize pre-trained knowledge more fully than the conventional PNN. Our proposal, an advanced variant of PNN, is called FPNN; it represents the combination of PNN with feature augmentation using bottleneck features [23], [24].

We demonstrate the proposed architecture in experiments on three recorded Japanese ASR tasks in which one is used as the source domain and the other two are used as target domains. We show that the lowest error rate is achieved by the proposed FPNN in both target domains. Moreover, experiments show that PNN-based approaches are robust to the order in which knowledge is sequentially transferred to the new domain.

II. RELATED WORK

Continual learning is another way to perform knowledge transfer [14], [18], [19]. In continual learning, models are continuously trained with new domain data while retaining the performance of the pre-trained domain. However, its success against real problems is doubtful because the model forgets the pre-trained knowledge. One continual learning approach is elastic weight consolidation (EWC) [18]. EWC can repeatedly re-train a model using new domain data while forgetting less of the pre-trained knowledge. Unfortunately, the new domain performance of EWC is worse than the same model trained with only the new domain data. PathNet [19] can utilize pre-trained knowledge for training a new domain model, but it needs many pre-trained models to cover various conditions as training involves combinations of parameters, the paths and weights across the models. Our final goal is to achieve continual learning that offers the best performance on both the new domain and the pre-trained domains. Although PNN-based approaches that can retain pre-trained knowledge are also regarded as continual learning, this paper focuses on improving the performance in the new domain.

III. APPROACHES FOR TRANSFER KNOWLEDGE

A. Pre-training and fine-tuning (PT/FT)

Pre-training and fine-tuning (PT/FT) [7], [8] is a very popular and simple approach. It is performed by using target domain data to fine-tune a pre-trained model. The fine-tuned model offers better performance in the target domain, because the initial parameters of PT/FT are better trained than the ones with random initialization.

B. Knowledge distillation (KD)

Knowledge distillation (KD) [10] is an approach for domain adaptation. We use the implementation presented in [12]. The KD approach uses two models: student and teacher models. The student model is trained using hard target y_i and soft target q_i. The hard target y_i is an element of one-hot vector $\boldsymbol{y}$ which has K dimensions corresponding to the number of output classes. If the i-th class is correct, y_i is 1, otherwise 0. The soft target q_i in $\boldsymbol{q}$ is a K dimension vector, the same as $\boldsymbol{y}$, and is computed by using the output (softmax) probability of the teacher model given by:

$$q_i = \frac{\exp(z_i(\boldsymbol{x})/T)}{\sum_{j=1}^{K}\exp(z_j(\boldsymbol{x})/T)}, \tag{1}$$

where $\boldsymbol{x}$ and $z_i(\boldsymbol{x})$ are the input feature and the pre-softmax output of the i-th class of the teacher model, respectively; T is temperature. As T becomes large, $\boldsymbol{q}$ approaches a uniform distribution. The student model is trained using the target labels of y_i and q_i so as to minimize the following loss function:

$$L = -(1-\rho)\sum_{i=1}^{K} y_i \log p_i(\boldsymbol{x}) - \rho\sum_{i=1}^{K} q_i \log p_i(\boldsymbol{x}), \tag{2}$$

where $p_i(\boldsymbol{x})$ is the output probability of the i-th class in the student model. The first and second terms represent cross entropy loss function of soft and hard targets, respectively; ρ is the weight of the hard and soft cross entropy losses.

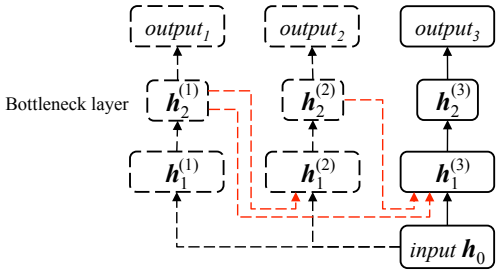

Fig. 2. *Architecture of three column feature augmented DNN (FDNN). Columns one and two on the left (dashed arrows) are trained on domain 1 and 2, respectively, and frozen while training on domain 3. Bottleneck features (red dashed arrows) of each pre-trained column are utilized for robustly training lower layers in the third column.*

C. Feature augmented DNN (FDNN)

Feature augmented DNN (FDNN) is simple approach and its architecture is described in Fig. 2. First and second columns from left side are trained on domain 1 and 2, respectively, and frozen while training on domain 3. We concatenate network input and bottleneck features [23], [24] derived from previous column DNNs when creating a new column DNN. In Fig. 2, red dashed arrows represent bottleneck features. The augmented input feature $\bar{\boldsymbol{h}}_0^{(n)}$ of each column DNN is described as follows:

$$\bar{\boldsymbol{h}}_0^{(n)} = \left[\boldsymbol{h}_0^{\top},\ \boldsymbol{h}_l^{(1:n-1)^{\top}}\right]^{\top}, \tag{3}$$

where l and n are the indices of layers and columns, respectively. $\bar{\boldsymbol{h}}_0^{(n)}$ is the super vector formed by concatenating the network input $\boldsymbol{h}_0$ and bottleneck features $\boldsymbol{h}_l^{(1:n-1)}$ in l-th layer of all pre-trained column DNNs from 1 to $n-1$.

The PT/FT approach can be applied for FDNN with the difference that the new column DNN parameters are copied

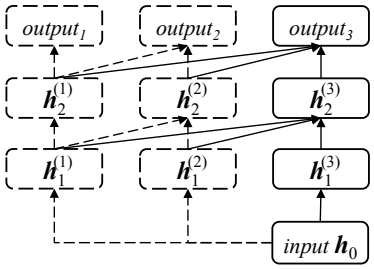

Fig. 3. *Depiction of a three column progressive neural network (PNN). Columns one and two on the left (dashed arrows) are trained on domain 1 and 2, respectively, and frozen while training on domain 3. The third column is added for the third domain, for which all pre-trained columns are leveraged.*

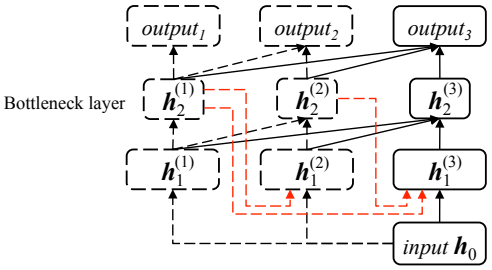

Fig. 4. *Architecture of three column feature augmented PNN (FPNN). The differences from PNN are shown by the red dashed arrows which represent bottleneck features. Bottleneck features are utilized to robustly train lower layers in the third column.*

from second to last hidden layers because the dimension of the input layer is different due to the concatenation with bottleneck features.

D. Progressive neural network (PNN)

The progressive neural network (PNN) [14] architecture consists of multi-column DNN as depicted in Fig. 3. In the PNN, the new column is added when transferring knowledge to a new domain. In other words, the number of columns corresponds to the number of domains. At the first step, the PNN is represented as a single column, whose structure is the same as the standard fully connected DNN. When transferring knowledge in support of a new domain, an additional column is concatenated to the pre-trained columns. In this case, the number of units and layers in the new column can be arbitrarily decided. The new column is trained with only the new domain data set the while pre-trained columns are frozen. Therefore, the PNN can be trained for the new domain without forgetting the knowledge in the previous domains. This paper does not use the adaptation layer presented in [14].

In each hidden layer, lateral connections from previous columns are leveraged for computing individual hidden activations. The hidden activation in the l-th layer of the n-th column, $\boldsymbol{h}_l^{(n)}$, is calculated as:

$$\boldsymbol{h}_l^{(n)} = f\left(\boldsymbol{W}_l^{(n)}\boldsymbol{h}_{l-1}^{(n)} + \sum_{m<n}\boldsymbol{V}_l^{(n:m)}\boldsymbol{h}_{l-1}^{(m)}\right), \quad (4)$$

where $l \leq L$, $m < n \leq N$ are the indices of layers and columns, respectively. $\boldsymbol{h}_l^{(n)} \in \mathbb{R}^{u_l^{(n)}}$ and $\boldsymbol{W}_l^{(n)} \in \mathbb{R}^{u_l^{(n)} \times u_{l-1}^{(n)}}$ are the hidden activation with $u_l^{(n)}$ being the number of units and the weight matrix of the l-th layer of the n-th column, respectively, $\boldsymbol{V}_l^{(n:m)} \in \mathbb{R}^{u_l^{(n)} \times u_{l-1}^{(m)}}$ are the weights for the lateral connections from the l-1-th layer of the m-th column to the l-th layer of n-th column; $\boldsymbol{h}_0$ is the network input, and the bias terms are omitted. f is a non-linear function e.g. sigmoid. The PNN approach makes the model parameters of the hidden layers larger when adding domains i.e. increasing "$\sum_l u_l^{(n)} \times u_{l-1}^{(n)} + \sum_l \sum_{m<n} u_l^{(n)} \times u_{l-1}^{(m)}$" with the addition of each new column.

In addition, the PNN can be combined with the PT/FT approach when the number of units and layers in a new column equals those in the pre-trained column. The operation is performed by initializing a new column using the previous column.

E. Feature augmented PNN (FPNN)

In [14], the lower layers of the newer domain were considered to have less influence on the output than that of the first column. We assume that the lower layers of the newer domain should also receive pre-trained knowledge if the parameters are randomly initialized. Based on the above intuition, we propose the new architecture, FPNN. It is composed of two techniques; PNN and feature augmentation using bottleneck features. Fig. 4 shows the FPNN trained with third domain data; the red dashed arrows represent bottleneck features. The difference between FPNN and PNN is the use of bottleneck features with the network input so as to improve the training of the lower layers. The augmented input feature is the same as FDNN in Eq. (3).

The PT/FT approach can be also applied for PFNN but the new column DNN parameters are copied from second to last hidden layers because the dimension of the input layer is different due to the concatenation of the bottleneck features.

IV. EXPERIMENTS

A. Experimental conditions

1) Data: To establish a common setup, we built and tested models using data generated from Japanese voice search commands for smartphone use. Three domain data sets (clean, car noise and distant talk) were used in the experiments. All data sets were recorded in a real room, car and distant talk environment, respectively. The SNR of car noise varied significantly (0 - 15 dB), while the distance in distant talk was 2.5m. The clean, car noise and distant talk data sets were split to yield 90, 70 and 60 hours for training, 10 hours for each development set, and one hour for each evaluation set. The input feature for all DNNs was 40 dimensional FBANK with the temporal context of 11 frames; dynamic features (Δ and $\Delta\Delta$) were used.

2) Common setup: All acoustic models shared the same configuration as regards the fully connected hidden layers and output layers. The nodes in the fully connected hidden layers and output contained 512 and 3072 nodes, respectively. The

TABLE I

CERs [%] for evaluation sets of clean, car noise and distant talk. The difference between model architectures in the 1st column is only the number of units per layer. Each model (6 models were built) was trained and evaluated using only target domain data. We argue that the number of model parameters, i.e. units, has no effect on the performance in each evaluation.

Model	Evaluation set CERs		
	clean	car noise	distant talk
Single(512)	9.0	21.6	14.7
Single(1024)	9.1	21.4	15.0

number of all DNN hidden layers was six and all bias terms were omitted. All architectures had a bottleneck layer, the fifth hidden layer, with 64 units. Only added target domain data was used in model retraining. The initial learning rate, 0.08, was halved if the frame accuracy of the development data was lower than the accuracy of the previous epoch. Training was terminated when the learning rate fell under 0.0008. As the optimizer, we used stochastic gradient descent (SGD) with a momentum value of 0.9 and a batch size of 128. We used Chainer [25] for DNN implementation and training. The language model was 3-gram and trained using a 1M Japanese web text corpus. For decoding, we used VoiceRex [26], [27]. We evaluated performance in terms of character error rate (CER).

3) Specific network architecture for each experiment: "Single" represents a model trained using only data from one domain with 512 or 1024 units per layer except the fifth layer which had 64 units. The parameters in "Single" were randomly initialized. In the KD approach, the parameters of temperature T and loss weight ρ were 3 and 0.1, respectively. The parameters of new column DNN and lateral connections of FDNN, PNN and FPNN were randomly initialized. We trained the models sequentially, i.e. starting from an initial model for domain 1, we create a model for domain 2. That model is then used as an initial model for training the model for domain 3. After training of each domain was completed, the parameters of PT/FT and student model of KD, called KD+PT/FT, were set at the corresponding parameters of the previous model. We also used the PT/FT approach to add new column DNN parameters for FDNN, PNN and FPNN (see the bottom of subsection III-C, III-D and III-E), called FDNN+PT/FT, PNN+PT/FT and FPNN+PT/FT respectively.

B. Results

Table I shows the results of Single DNN trained and evaluated with only each target domain data (clean, car noise and distant talk). The model architectures in 1st column are different only in the number of units per layer, that is 512 or 1024 units. The CER results in each domain are almost the same. We argue that the number of units per layer doesn't alter the CER because FDNN, PNN and FPNN increase the parameters due to the addition of column DNNs with new domains.

The 1st column in Table II shows approaches for transfer learning; we regard PT/FT as the baseline in this paper. The 2nd to 4th columns of Table II show the curriculum, the order in which knowledge is transferred to a new domain, and each evaluation set CERs, respectively. The 1st model parameters trained with clean were randomly initialized. In all approaches except KD+PT/FT approach, the targets of car noise and distant talk have better CERs than Single. These results confirm that the transfer learning approach is very important and effective for acoustic models. Only the KD+PT/FT approach does not work well as transfer learning because the KD+PT/FT approach yields a large model trained with various domains whereas the pre-trained model has little knowledge. The KD+PT/FT approach works well if the initial model is trained with a large amount of data [12]. FDNN and FDNN+PT/FT have better CERs than baseline. PNN CERs are better than the Single results but worse than those of PT/FT. Applying PT/FT to PNN improves the CERs of PNN+PT/FT significantly. We assume that PNN with PT/FT and feature augmentation can effectively supply pre-trained knowledge to a new column DNN because the initialized parameters in lower layers are better trained than the randomly initialized parameters. FPNN, which adds input bottleneck features to the network input of PNN, has better CERs than PT/FT so FPNN solves the above the problem. Moreover, of all approaches, FPNN+PT/FT attained the best CERs. Comparing FPNN+PT/FT with baseline, the relative error reductions for car noise and distant talk as target domains were 5.1% and 9.1%, respectively.

TABLE II

Curriculum and CERs [%] for evaluation sets of clean, car noise and distant talk from 2nd to 4th columns. The curriculum, which is the order in which existing knowledge is transferred to a new domain, is decided by the amount of training data. The 1st column indicates approaches for transfer learning and 1st model parameters of clean were randomly initialized.

Approach	Curriculum and evaluation set CERs		
	1. clean	2. car noise	3. distant talk
PT/FT	9.0	19.6	13.3
KD+PT/FT		22.6	13.5
FDNN		19.5	13.3
FDNN+PT/FT		19.3	13.1
PNN		21.4	14.2
PNN+PT/FT		18.8	12.6
FPNN		19.0	12.8
FPNN+PT/FT		**18.6**	**12.1**

TABLE III

Curriculum and CERs [%] for evaluation set of clean, car noise and distant talk from 2nd to 4th columns. The curriculum is decided by the Single CERs. We assume that FDNN-, PNN-, and FPNN-based architectures are less affected by the curriculum order of transfer knowledge than PT/FT.

Model	Curriculum and evaluation set CERs		
	1. clean	2. distant talk	3. car noise
PT/FT	9.0	13.9	21.0
KD+PT/FT		14.0	23.4
FDNN		13.5	19.4
FDNN+PT/FT		13.2	19.1
PNN		13.9	21.1
PNN+PT/FT		12.7	19.1
FPNN		12.7	19.2
FPNN+PT/FT		**12.3**	**18.7**

Table III also shows the curriculum and each evaluation set CER. In contrast to the experiments of Table II, here the order of domain training is different, i.e. we start with

clean domain, then distant talk and finally car noise. On the whole, the tendency of each evaluation set CER is worse than Table II which implies the importance of the amount of training data for the target domain. From the results of Table II and Table III, we argue that FDNN-, PNN-, and FPNN-based approaches are more robust than PT/FT to the curriculum used for transfer knowledge. This robustness is very important because the developers do not know the order in which data will be added in real problems. We assume that the transfer of fixed features in pre-trained knowledge to the target domain model contributes to the robustness to the order in which domain data is added. Relative to the baseline in Table III, FPNN+PT/FT achieved the significant reductions in relative error reductions for car noise and distant talk as target domains of 11.5% and 11.0%, respectively.

V. CONCLUSIONS

In this paper we proposed a novel acoustic model transfer learning approach based on progressive neural networks (PNN); it can offer the acquisition of new domain knowledge without forgetting the pre-trained knowledge. The proposed approach (FPNN) is composed of PNN and feature augmentation by bottleneck features so as to better leverage the pre-trained knowledge. Experiments on three real acoustic domains (clean, car noise, and distant talk) showed that FPNN could effectively transfer pre-trained knowledge to the new target domain and achieve lower error rates than the well-known approach of pre-training and fine-tuning (PT/FT), which fine-tunes a pre-trained model for the target domain. We also showed that PNN-based approaches are very robust to the curriculum i.e. the order in which existing knowledge is transferred to a new domain.

Future work includes model compression because PNN-based approaches make the model parameters larger when adding domains. We also plan to implement our proposal in other kinds of neural networks, such as convolutional neural networks and recurrent neural networks.

REFERENCES

[1] 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.

[2] Michael L. Seltzer, Dong Yu, and Yongqiang Wang, "An investigation of deep neural networks for noise robust speech recognition," in *Proc. of ICASSP*, 2013, pp. 7398–7402.

[3] K. Kinoshita, M. Delcroix, T. Yoshioka, T. Nakatani, E. Habets, R. Haeb-Umbach, V. Leutnant, A. Sehr, W. Kellermann, R. Maas, S. Gannot, and B. Raj, "The REVERB challenge: A common evaluation framework for dereverberation and recognition of reverberant speech," in *Proc. of WASPAA*, 2013, pp. 1–4.

[4] Li Deng, Geoffrey Hinton, and Brian Kingsbury, "New types of deep neural network learning for speech recognition and related applications: An overview," in *Proc. of ICASSP*, 2013, pp. 8599–8603.

[5] Jui-Ting Huang, Jinyu Li, Dong Yu, Li Deng, and Yifan Gong, "Cross-language knowledge transfer using multilingual deep neural network with shared hidden layers.," in *Proc. of ICASSP*, 2013, pp. 7304–7308.

[6] Van Hai Do, Nancy F. Chen, Boon Pang Lim, and Mark Hasegawa-Johnson, "Multi-task learning using mismatched transcription for under-resourced speech recognition.," in *Proc. of INTERSPEECH*, 2017, pp. 734–738.

[7] Amit Das and Mark Hasegawa-Johnson, "Cross-lingual transfer learning during supervised training in low resource scenarios," in *Proc. of INTERSPEECH*, 2015, pp. 3531–3535.

[8] Masayuki Suzuki, Ryuki Tachibana, Samuel Thomas, Bhuvana Ramabhadran, and George Saon, "Domain adaptation of CNN based acoustic models under limited resource settings," in *Proc. of INTERSPEECH*, 2016, pp. 1588–1592.

[9] Julius Kunze, Louis Kirsch, Ilia Kurenkov, Andreas Krug, Jens Johannsmeier, and Sebastian Stober, "Transfer learning for speech recognition on a budget," in *Proc. of 2nd Workshop on Representation Learning for NLP at the Annual Meeting of the ACL*, 2017, pp. 168–177.

[10] Geoffrey E. Hinton, Oriol Vinyals, and Jeffrey Dean, "Distilling the knowledge in a neural network," in *Proc. of NIPS Deep Learning and Representation Learning Workshop*, 2014.

[11] Austin Waters and Yevgen Chebotar, "Distilling knowledge from ensembles of neural networks for speech recognition," in *Proc. of INTERSPEECH*, 2016, pp. 3439–3443.

[12] Taichi Asami, Ryo Masumura, Yoshikazu Yamaguchi, Hirokazu Masataki, and Yushi Aono, "Domain adaptation of DNN acoustic models using knowledge distillation," in *Proc. of ICASSP*, 2017, pp. 5185–5189.

[13] Zhuo Chen Changliang Liu Xiong Xiao Guoli Ye Jinyu Li, Rui Zhao and Yifan Gong, "Developing far-field speaker system via teacher-student learning," in *Proc. of ICASSP*, 2018, pp. 5699–5703.

[14] Andrei A. Rusu, Neil C. Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell, "Progressive neural networks," *arXiv, 1606.04671*, 2016.

[15] Andrei A. Rusu, Matej Vecerik, Thomas Rothörl, Nicolas Heess, Razvan Pascanu, and Raia Hadsell, "Sim-to-real robot learning from pixels with progressive nets," *arXiv,1610.04286*, 2016.

[16] John Gideon, Soheil Khorram, Zakaria Aldeneh, Dimitrios Dimitriadis, and Emily Mower Provost, "Progressive neural networks for transfer learning in emotion recognition," in *Proc. of INTERSPEECH*, 2017, pp. 1098–1102.

[17] Sirui Xu and Eric Fosler-Lussier, "Application of progressive neural networks for multi-stream WFST combination in one-pass decoding," in *Proc. of ICASSP*, 2018, pp. 5914–5918.

[18] James Kirkpatrick, Razvan Pascanu, Neil C. Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell, "Overcoming catastrophic forgetting in neural networks," *arXiv, 1611.00796*, 2016.

[19] Chrisantha Fernando, Dylan Banarse, Charles Blundell, Yori Zwols, David Ha, Andrei A. Rusu, Alexander Pritzel, and Daan Wierstra, "PathNet: Evolution channels gradient descent in super neural networks," *arXiv,1701.08734*, 2017.

[20] Xugang Lu, Yu Tsao, Shigeki Matsuda, and Chiori Hori, "Ensemble modeling of denoising autoencoder for speech spectrum restoration," in *Proc. of INTERSPEECH*, 2014, pp. 885–889.

[21] Li Deng and John C. Platt, "Ensemble deep learning for speech recognition," in *Proc. of INTERSPEECH*, 2014, pp. 1915–1919.

[22] Jia Cui, Brian Kingsbury, Bhuvana Ramabhadran, George Saon, Tom Sercu, Kartik Audhkhasi, Abhinav Sethy, Markus Nussbaum-Thom, and Andrew Rosenberg, "Knowledge distillation across ensembles of multilingual models for low-resource languages," *in Proc. of ICASSP*, pp. 4825–4829, 2017.

[23] Frantisek Grézl, Martin Karafiát, Stanislav Kontar, and Jan Cernocký, "Probabilistic and bottle-neck features for LVCSR of meetings," in *Proc. of ICASSP*, 2007, pp. 757–760.

[24] Suyoun Kim, Bhiksha Raj, and Ian Lane, "Environmental noise embeddings for robust speech recognition," *arXiv, 1601.02553*, 2016.

[25] Seiya Tokui, Kenta Oono, Shohei Hido, and Justin Clayton, "Chainer: a next-generation open source framework for deep learning," in *Proc. of NIPS*, 2015.

[26] H. Masataki, D. Shibata, Y. Nakazawa, S. Kobashikawa, A. Ogawa, and K. Ohtsuki, "VoiceRex – spontaneous speech recognition technology for contact-center conversations," *NTT Technical Review*, vol. 5, no. 1, pp. 22–27, 2007.

[27] Takaaki Hori, Chiori Hori, Yasuhiro Minami, and Atsushi Nakamura, "Efficient WFST-based one-pass decoding with on-the-fly hypothesis rescoring in extremely large vocabulary continuous speech recognition," *IEEE Trans. Audio, Speech & Language Processing*, vol. 15, no. 4, pp. 1352–1365, 2007.