PARAMETER UNCERTAINTY FOR END-TO-END SPEECH RECOGNITION

Stefan Braun and Shih-Chii Liu

Institute of Neuroinformatics, University of Zurich and ETH Zurich, Switzerland

brauns@student.ethz.ch,shih@ini.ethz.ch

ABSTRACT

Recent work on neural networks with probabilistic parameters has shown that parameter uncertainty improves network regularization. Parameter-specific signal-to-noise ratio (SNR) levels derived from parameter distributions were further found to have high correlations with task importance. However, most of these studies focus on tasks other than automatic speech recognition (ASR). This work investigates end-to-end models with probabilistic parameters for ASR. We demonstrate that probabilistic networks outperform conventional deterministic networks in pruning and domain adaptation experiments carried out on the Wall Street Journal and CHiME-4 datasets. We use parameter-specific SNR information to select parameters for pruning and to condition the parameter updates during adaptation. Experimental results further show that networks with lower SNR parameters (1) tolerate increased sparsity levels during parameter pruning and (2) reduce catastrophic forgetting during domain adaptation.

Index Terms— end-to-end speech recognition, parameter uncertainty, pruning, adaptation

1. INTRODUCTION

Recently proposed end-to-end models for ASR [1, 2, 3, 4, 5, 6, 7, 8] present a significant simplification over DNN-HMM hybrids [9, 10] in both model architecture and training process. End-to-end models transcribe input speech to output text within a single neural network that is optimized in a single training stage. In contrast, hybrids consist of a combination of deep neural networks (DNNs) and hidden Markov models (HMMs) that are optimized disjointly in multiple training stages.

End-to-end models are *parametric* models: they provide tunable parameters for optimization, which correspond to the weights and biases of neural network units. Conventional end-to-end models use *deterministic* parameters, i.e. each parameter is a real value. While deterministic parameters encode the parameter magnitude, there is no direct encoding of the parameter uncertainty or the parameter importance to solve the task it was trained on. However, parameter importance is valuable information: potential use includes selective parameter pruning for saved computation or selective parameter fine-tuning to avoid catastrophic forgetting [11] in continual learning scenarios (e.g. speaker or domain adaptation).

Recent work has explored parameter uncertainty in neural networks by encoding parameters in a *probabilistic* fashion [12, 13, 14]. Probabilistic neural networks sample parameters from probability distributions learned on training data such that each parameter exhibits a learned degree of uncertainty. The inherent parameter uncertainty makes probabilistic networks less sensitive to parameter perturbations and less prone to overfitting [12, 14]. The relation between the magnitude of a parameter and its uncertainty allows one to establish parameter-specific SNR levels. Previous studies show that there is a high correlation between parameter SNR and parameter importance as demonstrated in pruning experiments for tasks other than ASR [12, 13]. To the best of our knowledge, only one study investigated probabilistic neural networks for end-to-end ASR [14]. The probabilistic network was derived in a variational inference framework from a Bayesian perspective. The evaluation was carried out using a single probabilistic network on the TIMIT dataset with a focus on parameter pruning.

This work proposes an alternative derivation of probabilistic networks from a parameter perspective, without requiring a Bayesian interpretation of the model. We also extend the set of ASR tasks to include domain adaptation from clean speech to noisy speech. In order to prevent forgetting of the original task, we propose the use of a SNR-based regularization scheme to condition parameter updates on parameter importance. Furthermore, this work evaluates probabilistic networks with distinct SNR levels. We compare how networks with different SNR levels tolerate pruning of parameters and how the level of catastrophic forgetting changes across these networks during domain adaptation.

2. PROBABILISTIC END-TO-END MODELS

2.1. Random variable parameters

We consider end-to-end models for ASR that transcribe speech input to text output with a single neural network. A conventional end-to-end model consists of a set of $n = 1, ..., N$ *deterministic* parameters $\theta = \{\theta_1, ..., \theta_N\}$ which represent the weights and biases of the neural network units. In this work, we consider *probabilistic* parameters $\Theta = \{\Theta_1, ..., \Theta_N\}$ for our model. While there are many possible ways to define the random variables Θ_n, we choose a Gaussian distribution such that $\Theta \sim \mathcal{N}(\mu, \sigma)$. Note that every parameter Θ_n is described with a parameter-specific mean μ_n and standard deviation σ_n. The mean and standard deviation represent the expected parameter value and its uncertainty.

Similar to related work [12, 13, 14], we define a parameter-specific signal-to-noise ratio:

$$\mathrm{SNR}_n = \frac{|\mu_n|}{\sigma_n} \tag{1}$$

The results of previous studies [12, 13, 14] imply that SNR levels can be used to identify the important parameters that are useful for solving a task.

2.2. Training with random variables

The training process optimizes both the mean $\mu \in \mathbb{R}^N$ and standard deviation $\sigma \in \mathbb{R}^N$ of the $n = 1, ..., N$ network parameters. The standard deviation σ is parameterised with the proxy parameter $\beta \in \mathbb{R}^N$ and the softplus function [15] $\sigma = \log(\exp(\beta) + 1)$ to ensure

that σ is positive. We use a training procedure similar to [12] that updates network parameters as described in Eq. (2)-(5):

$$\epsilon \sim \mathcal{N}(0, 1) \tag{2}$$

$$\theta = \mu + \log(\exp(\beta) + 1) \cdot \epsilon \tag{3}$$

$$\mathcal{L} = f(\theta, x, y) \tag{4}$$

$$\mu', \beta' \leftarrow \text{optimizer}(\mu, \beta, \nabla\mathcal{L}_\mu, \nabla\mathcal{L}_\beta) \tag{5}$$

The following procedure is repeated for every mini batch: First, noise samples $\epsilon \in \mathbb{R}^N$ are drawn from a standard normal distribution (Eq. (2)). The noise samples are scaled by σ and shifted by μ to compute the parameters $\theta \in \mathbb{R}^N$ (Eq. (3)). For the forward pass, the parameters θ and the network input x and target labels y are used to compute the loss $\mathcal{L}$ (Eq. (4)). For the backward pass, the gradients $\nabla\mathcal{L}_\mu, \nabla\mathcal{L}_\beta$ are computed and the parameters μ, β are updated with an optimizer (Eq. (5)).

The first two steps described in Eq. (2) and (3) are referred to as the "reparameterization trick" in the literature [16] and yield the same effect as sampling θ from $\mathcal{N}(\mu, \beta)$, but they keep the sampling operation differentiable wrt. μ, β. Training with probabilistic parameters is not different from training with deterministic parameters with respect to the loss function (Eq. (4)) and the optimizer (Eq. (5)). Note that the loss function and the optimizer can be of arbitrary choice.

2.3. Related work

Probabilistic neural network models have been mostly explored from a variational inference perspective [12, 13, 14, 17]. The embedding in a Bayesian framework allows these approaches to introduce additional loss terms that are interpreted as parameter complexity cost terms [12, 13] or minimum description length cost terms [14, 17]. In contrast, this work develops probabilistic networks from a parameter-based perspective, and we do not develop additional loss terms by a Bayesian interpretation of the network.

To the best of our knowledge, only one study evaluated probabilistic networks for end-to-end ASR [14]. This study used a long short-term memory (LSTM)-based Connectionist Temporal Classification (CTC) model with 140k parameters on the 5h TIMIT dataset. The effect of parameter pruning was evaluated for a single probabilistic network, and no adaptation scenario was considered. In contrast, this study evaluates multiple probabilistic networks with different SNR levels for both pruning and domain adaptation scenarios. Even the non-ASR studies investigated only SNR-based pruning, but no continual learning scenarios such as domain adaptation.

Probabilistic networks with Gaussian parameters are closely related to the weight noise regularizer for recurrent neural networks (RNNs) [18]. This regularizer adds noise from a normal distribution $\mathcal{N}(0, \sigma_g)$ to the network parameters before the forward pass. The standard deviation σ_g is a single scalar hyperparameter used for all network parameters and is not updated during training [19, 20]. In contrast, we use a separate standard deviation per parameter and perform gradient-based updates on the standard deviation in training.

3. EXPERIMENTAL SETUP

3.1. Datasets

All experiments are carried out as ASR tasks on the Wall Street Journal (WSJ) [21] and CHiME-4 [22] datasets presented in Table 1. The WSJ dataset provides single channel read speech data recorded in *clean* conditions. The CHiME-4 dataset provides read speech data recorded from a 6-channel tablet in *noisy* conditions (bus, street junction, cafe and pedestrian area). In this work, we only use the channel 5 data that leads to the lowest error rates on CHiME-4.

Dataset	Subset	# hours	Comment
WSJ	train_si284	81.0	-
WSJ	test_dev93	1.0	-
WSJ	test_eval92	0.7	-
CHiME-4	tr05_simu_real	18.0	only CH5
CHiME-4	dt05_real	2.5	only CH5
CHiME-4	et05_real	2.0	only CH5

Table 1. Datasets used for experimentation.

The audio data was pre-processed into 123-dimensional filterbank features (25ms frames, 10ms frame shift, 40 Mel-spaced filterbanks, energy coefficient, 1st and 2nd order delta) and normalized to zero-mean and unit-variance per sample. Both the WSJ and CHiME-4 datasets use the same alphabet of 59 units (characters, digits etc.) as output labels which were obtained with the EESEN preprocessing routines [3]. The character error rate (CER) is used as the performance metric.

3.2. Model architecture

All models share the same basic architecture: 5 layers of bidirectional LSTMs [23] with 320 units in each direction and a final 640x59 projection to the output labels. The deterministic models use default LSTM units and consist of a parameter set θ_D with LSTM weights w^{LSTM}, biases b^{LSTM} and projection weights w^{PROJ} (Eq. (6), ∼11M parameters).

$$\theta_D = \{w^{LSTM}, b^{LSTM}, w^{PROJ}\} \tag{6}$$

The probabilistic models use LSTMs with Gaussian weights and consist of a parameter set θ_P with LSTM weight means μ^{LSTM}, parameterised weight standard deviations β^{LSTM}, biases b^{LSTM} and projection weights w^{PROJ} (Eq. (7), ∼22M parameters).

$$\theta_P = \{\mu^{LSTM}, \beta^{LSTM}, b^{LSTM}, w^{PROJ}\} \tag{7}$$

We initialize w^{LSTM}, μ^{LSTM} and w^{PROJ} with the Xavier uniform initialization [24] according to Eq. (8). The same random seed is used to ensure that probabilistic and deterministic models start training with identical weight and weight mean, i.e. $w^{LSTM} = \mu^{LSTM}$. All biases b^{LSTM} are initialized to 0. The parameterized standard deviation β^{LSTM} is initialized according to Eq. (9). This initialization results in an average SNR of 1.0 for the LSTM weight parameters. We find that this SNR value gives the same convergence speed during training as a network that uses the deterministic parameters. Networks with lower SNR initializations take longer to converge due to their high noise level.

$$A_{ij} \sim \mathcal{U}\left(-\frac{\sqrt{6}}{i+j}, \frac{\sqrt{6}}{i+j}\right), A \in \mathbb{R}^{i \times j} \tag{8}$$

$$B_{ij} = \log\left(\exp(\frac{1}{2}\frac{\sqrt{6}}{i+j}) - 1)\right), B \in \mathbb{R}^{i \times j} \tag{9}$$

3.3. Baseline training

The baseline deterministic models `D/WSJ`, `D/CHiME` and `D/MIX` use deterministic LSTM units (Eq. (6)); and are trained on train_si284,

tr05_simu_real and the combined train_si284 + tr05_simu_real subsets respectively. The probabilistic models P/WSJ/λ_β use LSTMs with Gaussian weights (Eq. (7)) and are trained on train_si284. All models are trained with the CTC loss function $\mathcal{L}_{CTC}$ [25] and the Adam optimizer (learning rate 1e-3) [26] for 25 epochs, and the model from the epoch with the lowest CER on test_dev93 is selected for evaluation. For the probabilistic models, we enforce lower SNR parameters by using weight decay $\mathcal{L}_\beta = ||\beta||_2^2$ on the parameterized weight standard deviation β^{LSTM}. The decay term L_β is scaled by the factor λ_β such that the complete loss function is $\mathcal{L} = \mathcal{L}_{CTC} + \lambda_\beta \mathcal{L}_\beta$. We tune the hyperparameter λ_β by a grid search in the range 1e-7 to 1e-5. Three models with $\lambda_\beta = 0$, 1e-6 and 4e-6 are selected for evaluation. The model P/WSJ/0 is the baseline probabilistic model, the model P/WSJ/1e-6 achieves the lowest CER of all probabilistic models and the model P/WSJ/4e-6 achieves similar CER compared to P/WSJ/0 but with lower SNR.

3.4. Testing

All our models are tested with strict end-to-end criteria and without the use of external language models. The CTC output is decoded in a greedy fashion: at every time step, the label with the highest probability is selected. The probabilistic models are tested with the mean weights μ^{LSTM}, i.e. the LSTM parameters are not sampled during testing. The interested reader is referred to related work for experiments that explore parameter sampling during testing [12].

3.5. Pruning

The pruning experiment is carried out on the models that were trained on train_si284, i.e. D/WSJ, P/WSJ/0, P/WSJ/1e-6 and P/WSJ/4e-6. The LSTM weight parameters, which account for $> 99\%$ of the model parameters, are pruned while the rest of the model parameters is left unchanged. For the deterministic models, the LSTM weights w^{LSTM} are ordered by magnitude and the lowest X percent of magnitude weights are pruned, i.e. set to zero. For the probabilistic models, the LSTM mean weights μ^{LSTM} are ordered by SNR and the lowest X percent of SNR weight means are pruned. The models are tested on test_dev93 without any retraining after pruning. The probabilistic models use the LSTM mean weights μ^{LSTM} for testing.

3.6. Adaptation

The adaptation experiment is carried out on the following models that were trained on train_si284: D/WSJ, P/WSJ/0, and P/WSJ/4e-6. The models were originally trained on clean speech data from WSJ, and now they are adapted to noisy speech data from the CHiME-4 dataset by further training on the dt05_real subset. The models are adapted for 25 epochs with the CTC loss $\mathcal{L}_{CTC}$ and the Adam optimizer (learning rate 1e-3). Note that we use the same number of epochs and learning rate for adaptation as during baseline training. This strategy is different from conventional adaptation setups that use fine-tuning with fewer epochs and smaller learning rates for adaptation (e.g. [27]). In order to analyze the effect of Gaussian weights, we only adapt the weights w^{LSTM} (deterministic models) or the weight mean μ^{LSTM} (probabilistic models) of the LSTM cells. The biases b^{LSTM} and the projection weights w^{PROJ} are left unchanged. For the deterministic model, we propose an auxiliary L2 penalty $\mathcal{L}_{L2}$ between updated weight value w^{LSTM} and pre-adaptation weight value w^{LSTM*}:

$$\mathcal{L}_{L2} = (w^{LSTM} - w^{LSTM*})^2 \tag{10}$$

Model	WSJ test eval92	WSJ test dev93	CHiME-4 et05 real	CHiME-4 dt05 real
D/MIX	**6.1**	**8.5**	**33.4**	**21.9**
D/CHiME	17.1	21.7	37.2	26.6
D/WSJ	6.5	8.9	57.7	45.3
P/WSJ/0	6.4	8.8	56.6	44.5
P/WSJ/1e-6	**6.1**	**8.5**	55.9	43.4
P/WSJ/4e-6	6.4	9.0	55.9	44.0
Single-E2E [28]	-	-	40.9	29.5
ESPnet [29]	7.6	10.1	-	-

Table 2. Baseline CER [%] results for clean speech (WSJ) and noisy speech (CHiME-4). The lowest CER on each subset is printed bold.

The L2 penalty prevents catastrophic forgetting by forcing weight updates to stay close to the original value. For the probabilistic models, we include an auxiliary SNR penalty $\mathcal{L}_{SNR}$ between updated mean weight value μ^{LSTM} and pre-adaptation mean weight value μ^{LSTM*}:

$$\mathcal{L}_{SNR} = \text{SNR}(\mu^{LSTM} - \mu^{LSTM*})^2 \tag{11}$$

The inclusion of the SNR value in the loss term penalizes updates on parameters with higher SNR, which are assumed to be more important than lower SNR parameters to solve the original clean speech task. Note that besides the use of SNR information, the SNR penalty from Eq. (11) is similar to the L2 penalty from Eq. (10). The auxiliary penalties are scaled with the parameter λ_{aux} that is varied between $\{0, 0.1, ..., 727.9, 1000.0\}$ in 30 geometrically spaced steps, and the full loss function for adaptation is $\mathcal{L} = \mathcal{L}_{CTC} + \lambda_{aux}\mathcal{L}_{aux}$.

4. EXPERIMENTAL RESULTS

4.1. Baselines

The baseline evaluation results on the WSJ and CHiME-4 test and development sets are reported in Table 2. The deterministic model D/MIX trained on both clean speech (WSJ) and noisy speech (CHiME-4) achieves the lowest CER across all evaluation scenarios. The other models are trained on either clean speech or noisy speech, and they only achieve low error rates in the same noise conditions they were trained on.

When considering only models trained on clean speech (WSJ), the probabilistic models P/WSJ/0 and P/WSJ/4e-6 perform on par with the deterministic model D/WSJ. The model P/WSJ/1e-6 performs best and achieves up to 6.2% relative CER reduction compared to D/WSJ.

Recent work on end-to-end models with deterministic weights and without external language models reports similar error rates. The models from recent work are represented by Single-E2E [28] (trained on tr05_simu_real, channel 5) and ESPnet [29] (trained on train_si284).

4.2. SNR statistics

We compute the SNR statistics on the Gaussian LSTM weights of the probabilistic models after training and report the results in Table 3. Before training, all models were initialized with the same median SNR level of 1.0 according to Eq. (9). After training, the

Model	Median	Mean±Std	Min	Max
P/WSJ/0	2.9	3.6±2.9	2.7e-6	80.1
P/WSJ/1e-6	1.9	2.3±1.9	4.1e-7	49.6
P/WSJ/4e-6	1.0	1.3±1.1	1.4e-7	41.8

Table 3. SNR statistics for the LSTM weights of the probabilistic models obtained after a completed training on train_si284. The statistics are computed over SNR $= |\mu^{LSTM}|/\sigma^{LSTM}$.

models show different SNR levels: P/WSJ/4e-6 still shows a median SNR of 1.0, while the model P/WSJ/0 increases the median SNR up to 2.9. This indicates that the SNR level is indeed controllable by the additional cost term $\mathcal{L}_\beta$ on the parameterized standard deviation β^{LSTM}. Interestingly, both models achieve similar CER during evaluation despite significantly different noise levels.

4.3. Pruning

The pruning results are reported in Figure 1. All probabilistic models are able to achieve lower error rates with the same sparsity level than the deterministic model D/WSJ when the sparsity is between 50% to 90%. Probabilistic models with lower SNR tolerate higher sparsity levels than models with higher SNR, and P/WSJ/4e-6 tolerates the highest sparsity levels. With a 75% sparsity level, P/WSJ/4e-6 achieves 11.3% CER, which is a relative CER reduction of 58.9% compared to the 27.5% CER of D/WSJ.

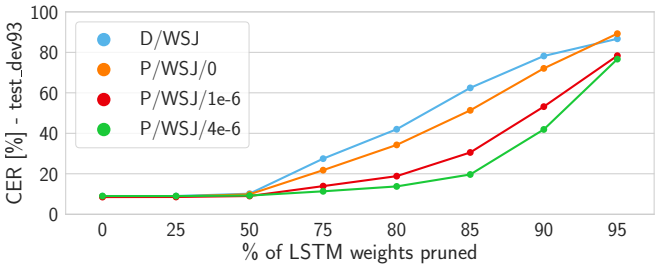

Fig. 1. Weight pruning results when testing on the WSJ test_dev93 subset. All probabilistic models show lower error rates at similar sparsity levels than the deterministic model D/WSJ. The probabilistic model with the lowest SNR P/WSJ/4e-6 shows the smallest error rate increase under pruning.

4.4. Adaptation

We evaluate the CER of the adapted models for every epoch of adaptation on the subsets test_eval92 (clean speech) and et05_real (noisy speech) and report the results in Figure 2. The results show three tiers of adaptation characteristics. The trade-off between error rates on clean speech and noisy speech is smallest for the lower SNR model P/WSJ/4e-6, intermediate for the higher SNR model P/WSJ/0 and highest for the deterministic model D/WSJ. In other words, D/WSJ is more prone to forgetting the original clean speech task than both probabilistic models, and a lower parameter SNR further reduces forgetting. When allowing for 8.0% CER on test_eval92, then P/WSJ/4e-6 reaches 42.7% CER on et05_real, while D/WSJ reaches 50.2% CER, a relative reduction of 15%.

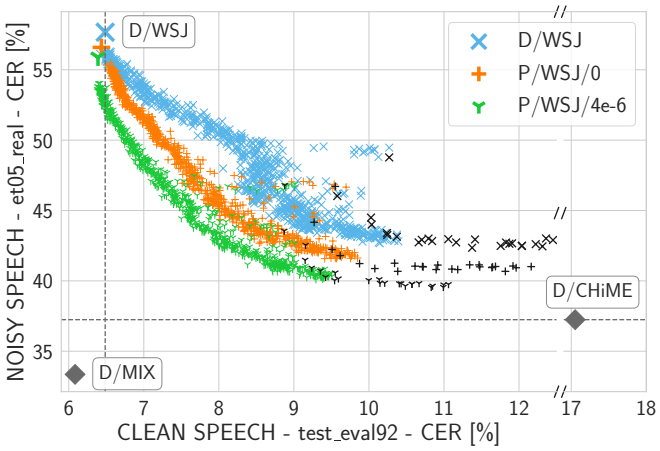

Fig. 2. Adaptation results when adapting networks trained on clean speech, to noisy speech. Large dots show pre-adaptation error rates. Black dots denote models adapted with $\lambda_{aux} = 0$ (no additional penalty), while colored dots correspond to models with $\lambda_{aux} > 0$ (L2 or SNR penalty active). Every dot represents a different epoch of adaptation and a different λ_{aux}. The probabilistic networks show less forgetting on the original clean speech data during adaptation. The low SNR probabilistic model P/WSJ/4e-6 shows the least amount of forgetting.

5. CONCLUSION

In this work we evaluated end-to-end models for ASR that use LSTM units with probabilistic weight parameters. The parameters are sampled from a Gaussian distribution with a parameter-specific mean and standard deviation. Despite the probabilistic formulation, the model is trainable with the same cost function as a model with deterministic parameters.

Experimental results show that probabilistic models achieved error rates on par or better than deterministic models. When pruning weights on an already trained model, the probabilistic models tolerated higher sparsity levels at lower error rates than the deterministic models. Also, during an adaptation experiment from clean to noisy speech, the probabilistic models showed less forgetting on the original clean speech task than deterministic models. A key advantage of probabilistic models is the availability of the parameter-specific SNR, which is highly correlated with the importance of a parameter for the task it was trained on. The parameter-specific SNR helped to identify less important parameters for pruning and to restrict updates on important parameters during adaptation.

The average SNR of the end-to-end model parameters is controllable by an additional loss term that enforces higher standard deviation. When comparing probabilistic models with different SNR levels, our results show that models with lower SNR exhibit improved pruning and adaptation characteristics. Future studies include evaluating probabilistic neural networks on larger speech datasets; and using acoustic models that are based on neural network units other than LSTMs, e.g. convolutional neural networks (CNNs).

6. ACKNOWLEDGEMENTS

This work was partially supported by the European Union's Horizon 2020 research and innovation program under grant agreement No 644732 and the Samsung Advanced Institute of Technology.

7. REFERENCES

[1] A. Graves and N. Jaitly, "Towards End-To-End Speech Recognition with Recurrent Neural Networks," in *Proceedings of the 31th International Conference on Machine Learning (ICML)*, 2014, pp. 1764–1772.

[2] A. Hannun et al., "Deep speech: Scaling up End-to-End Speech Recognition," *arXiv preprint arXiv:1412.5567*, 2014.

[3] Y. Miao, M. Gowayyed, and F. Metze, "EESEN: End-to-end Speech Recognition using Deep RNN Models and WFST-based Decoding," in *IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, 2015, pp. 167–174.

[4] J. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio, "Attention-Based Models for Speech Recognition," in *Advances in Neural Information Processing Systems 28 (NIPS)*, 2015, pp. 577–585.

[5] D. Amodei et al., "Deep speech 2: End-to-End Speech Recognition in English and Mandarin," in *Proceedings of the 33rd International Conference on Machine Learning (ICML)*, 2016, pp. 173–182.

[6] W. Chan, N. Jaitly, Q. Le, and O. Vinyals, "Listen, Attend and Spell: A Neural Network for Large Vocabulary Conversational Speech Recognition," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2016, pp. 4960–4964.

[7] L. Lu, X. Zhang, and S. Renals, "On Training the Recurrent Neural Network Encoder-Decoder for Large Vocabulary End-to-End Epeech Recognition," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2016, pp. 5060–5064.

[8] S. Kim, T. Hori, and S. Watanabe, "Joint CTC-Attention based End-to-End Speech Recognition using Multi-task Learning," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2017, pp. 4835–4839.

[9] H. Bourlard and N. Morgan, *Connectionist Speech Recognition: A Hybrid Approach*, Kluwer Academic Publishers, 1994.

[10] G. Hinton et al., "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.

[11] M. McCloskey and N. J. Cohen, "Catastrophic Interference in Connectionist Networks: The Sequential Learning Problem," in *Psychology of learning and motivation*, vol. 24, pp. 109–165. Elsevier, 1989.

[12] C. Blundell, J. Cornebise, K. Kavukcuoglu, and D. Wierstra, "Weight Uncertainty in Neural Networks," in *Proceedings of the 32nd International Conference on Machine Learning (ICML)*, 2015, pp. 1613–1622.

[13] M. Fortunato, C. Blundell, and O. Vinyals, "Bayesian Recurrent Neural Networks," *arXiv preprint arXiv:1704.02798*, 2017.

[14] A. Graves, "Practical Variational Inference for Neural Networks," in *Advances in Neural Information Processing Systems 24 (NIPS)*, 2011, pp. 2348–2356.

[15] C. Dugas, Y. Bengio, F. Bélisle, C. Nadeau, and R. Garcia, "Incorporating Second-Order Functional Knowledge for Better Option Pricing," in *Advances in Neural Information Processing Systems 13 (NIPS)*, 2001, pp. 472–478.

[16] D. P. Kingma, T. Salimans, and M. Welling, "Variational Dropout and the Local Reparameterization Trick," in *Advances in Neural Information Processing Systems 28 (NIPS)*, 2015, pp. 2575–2583.

[17] G. E. Hinton and D. Van Camp, "Keeping the Neural Networks Simple by Minimizing the Description Length of the Weights," in *Proceedings of the sixth annual conference on Computational learning theory (COLT)*, 1993, pp. 5–13.

[18] K.-C. Jim, C. L. Giles, B. G. Horne, et al., "An Analysis of Noise in Recurrent Neural Networks: Convergence and Generalization," *IEEE Transactions on neural networks*, vol. 7, no. 6, pp. 1424–1438, 1996.

[19] E. Battenberg et al., "Exploring Neural Transducers for End-to-End Speech Recognition," in *IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, 2017, pp. 206–213.

[20] M. Ravanelli, P. Brakel, M. Omologo, and Y. Bengio, "Light Gated Recurrent Units for Speech Recognition," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 2, no. 2, pp. 92–102, 2018.

[21] J. Garofalo, D. Graff, D. Paul, and D. Pallett, "CSR-I (WSJ0) complete, LDC93S6A," *Linguistic Data Consortium, Philadelphia*, 2007.

[22] 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 & Language*, 2016.

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

[24] X. Glorot and Y. Bengio, "Understanding the Difficulty of Training Deep Feedforward Neural Networks," in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2010, pp. 249–256.

[25] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist Temporal Classification: Labelling unsegmented Sequence Data with Recurrent Neural Networks," in *Proceedings of the 23rd International Conference on Machine learning (ICML)*, 2006, pp. 369–376.

[26] D. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[27] T. Ochiai, S. Watanabe, S. Katagiri, T. Hori, and J. R. Hershey, "Speaker Adaptation for Multichannel End-to-End Speech Recognition," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018, pp. 6707–6711.

[28] S. Kim and I. R. Lane, "Recurrent Models for Auditory Attention in Multi-Microphone Distant Speech Recognition," in *Interspeech*, 2016, pp. 3838–3842.

[29] S. Watanabe, T. Hori, S. Karita, T. Hayashi, J. Nishitoba, Y. Unno, N. E. Y. Soplin, J. Heymann, M. Wiesner, N. Chen, et al., "ESPnet: End-to-End Speech Processing Toolkit," in *Interspeech*, 2018, pp. 2207–2211.