Interspeech 2018
2-6 September 2018, Hyderabad, India

Multi-task Learning with Augmentation Strategy for Acoustic-to-word Attention-based Encoder-decoder Speech Recognition

Takafumi Moriya[1], *Sei Ueno*[1,2], *Yusuke Shinohara*[1],
Marc Delcroix[3], *Yoshikazu Yamaguchi*[1], *Yushi Aono*[1]

[1]NTT Media Intelligence Laboratories, NTT Corporation, Japan
[2]Graduate School of Informatics, Kyoto University, Japan
[3]NTT Communication Science Laboratories, NTT Corporation, Japan
takafumi.moriya.nd@hco.ntt.co.jp, ueno@sap.ist.i.kyoto-u.ac.jp

Abstract

In this paper, we propose a novel training strategy for attention-based encoder-decoder acoustic-to-word end-to-end systems. Accuracy of end-to-end systems has greatly improved thanks to careful tuning of model structure and the introduction of novel training strategies to stabilize training. For example, multi-task learning using a shared-encoder is often used to escape from bad local optima. However, multi-task learning usually relies on a linear interpolation of the losses for each sub-task and consequently, the shared-encoder is not optimized for each task. To solve the above problem, we propose a multi-task learning with augmentation strategy. We augment the training data by creating multiple copies of the original training data to suit different output targets associated with each sub-task. We use each target loss sequentially to update the parameters of the shared-encoder so as to enhance the versatility of capturing acoustic features. This strategy enables better learning of the shared-encoder as each task is trained with a dedicated loss. The parameters of the word-decoder are jointly updated via the shared-encoder when optimizing the word prediction task loss. We evaluate our proposal on various speech data sets, and show that our models achieve lower word error rates than both single-task and conventional multi-task approaches.

Index Terms: end-to-end speech recognition, attention model, multi-task learning

1. Introduction

Automatic speech recognition (ASR) systems have been dramatically improved with the introduction of deep neural networks (DNN) [1]. Currently, a typical ASR system is composed of several modules including acoustic, lexicon, and language models. The best system for the conversational telephone speech task [2] adopts the approach of combining deep learning and probabilistic models [3]. Such ASR systems can now match human recognition performance [4, 5], but at the cost of large runtime latency and less portability. Recent ASR systems overcomes this problem by mapping acoustic feature sequences to word sequences directly; this is called acoustic-to-word end-to-end speech recognition.

End-to-end speech recognition systems are based on the simple recurrent neural network (RNN) with long short-term memory (LSTM) architecture without requiring latent state transition models such as Hidden Markov Models (HMMs). There are two major approaches: connectionist temporal classification (CTC) [6–9] marginalizes and condenses all possible frame-wise output symbol sequences, while the encoder-decoder model with an attention mechanism [10–14], first encodes the acoustic feature input into an intermediate representation with one RNN and then decodes it into a target symbol sequence with another RNN. However, many end-to-end systems use still subword units, such as phonemes, syllables and characters. Therefore, to achieve optimal performance theses still depend on a pronunciation lexicon and language model [15], and only slow decoding is possible.

In this paper, we deal with end-to-end systems that outputs whole words from audio features without requiring an external language model. Several studies have attempted to build acoustic-to-word models [16–21]. To build such systems, it is essential to include other end-to-end systems that output phonemes and subwords with high accuracy and set the initialized parameters carefully. Various training strategies have been developed to regularize the training and prevent being trapped in local optima, such as pre-training (PT) and multi-task learning (MT) with linear interpolation (IP) [16, 19–24]. The MT approach is often used to optimize the encoder shared by CTC (sub-task) and attention-decoder (main-task), and has been shown to be superior to single-task learning (ST). Although the MT approach can make the shared-encoder attain better intermediate representations such that the decoder can achieve lower error rates, the shared-encoder is not directly optimized for each task due to the averaging of the multi-task losses. We consider that the shared-encoder should be explicitly updated for each task loss to enhance the versatility of capturing acoustic features.

To address this problem, we propose a data augmentation strategy (AS) for multi-task learning. Instead of augmenting the data by applying various kinds of signal distortions to the input acoustic features as is often done, our idea is to augment the training data by creating multiple copies with different labels that reflect the corresponding output targets of each task. A similar idea has been recently used in knowledge distillation [25]. They applied the AS to the multi-teacher student approach where update is performed sequencially in each computation of each teacher loss instead of using linear interpolated losses; achieving better performance than models trained from scratch or using interpolated loss functions.

In this paper, we propose a MT with AS for encoder-decoder models. Our proposal uses each target loss sequentially and updates the parameters of the shared-encoder for each loss per minibatches. This strategy enables better learning of the shared-encoder as each task is trained with a dedicated loss, which is unaffected by other task losses. Experiments show the efficacy of MT with AS and its superiority to ST and MT with IP using various recorded Japanese speech tasks including call center dialogue and voice search (44 to 1900 hours).

10.21437/Interspeech.2018-1866

2. End-to-end ASR models

We explain the two basic approaches to end-to-end speech recognition. $\boldsymbol{X} = (\boldsymbol{x}_1, ..., \boldsymbol{x}_T)$ and $\boldsymbol{y} = (y_1, ..., y_L)$ are the input acoustic feature sequence of length-T and target label sequence of length-L, respectively, where $y_l \in \{1, ..., K\}$; K is the number of target labels. In this paper, we use characters for CTC and words in the attention-based model as the targets.

2.1. Connectionist Temporal Classification (CTC)

CTC deals with the target labels and the extra "blank" label, ϕ, and learns the mapping between sequences of different lengths. By inserting a blank label between two consecutive labels and allowing each label to be repeated, label sequence $\boldsymbol{y}$ can be expanded to a set of length-T sequences $\Omega(\boldsymbol{y})$. Inversely, each CTC path $\boldsymbol{\pi} \in \Omega(\boldsymbol{y})$ with redundancy can be reduced to the original label sequence $\boldsymbol{y}$ after removing all repeating labels and blank labels, where $\boldsymbol{\pi} = (\pi_1, ..., \pi_T)$ and $\pi_t \in \{1, ..., K\} \cup \{\phi\}$.

CTC loss is defined using the probabilities of all CTC paths included in $\Omega(\boldsymbol{y})$ as indicated by:

$$p(\boldsymbol{y}|\boldsymbol{X}) = \sum_{\pi \in \Omega(\boldsymbol{y})} p(\pi|\boldsymbol{X}) = \sum_{\pi \in \Omega(\boldsymbol{y})} \prod_{t=1}^{T} p(\pi_t|\boldsymbol{x}_t), \quad (1)$$

where the posterior probabilities $p(\pi_t|\boldsymbol{x}_t)$ are calculated by a multi-layer bidirectional RNN. The CTC loss and its gradient with respect to the network parameters are efficiently computed with the forward-backward algorithm. CTC-based models do not explicitly learn the internal relationship between labels since they assume that the probabilities of each label are independent of others.

2.2. Attention-based encoder-decoder model

The other end-to-end ASR approach is the attention-based encoder-decoder architecture [10–14]. This architecture is composed of two distinct subnetworks. One is the encoder subnetwork; it transforms an acoustic feature sequence into an intermediate representation whose length corresponds to the length of the input sequence, T. The other is the decoder subnetwork; it predicts a label sequence from the intermediate information provided by the encoder subnetwork. Decoded label length, L, is usually less than input feature length, T. The decoder uses only the relevant portion of the encoded sequential representations for predicting a label at each time step using the attention mechanism.

The attention-based model is formulated as follows. The encoder transforms $\boldsymbol{X}$ into intermediate representation vectors $\boldsymbol{H} = (\boldsymbol{h}_1, ..., \boldsymbol{h}_T)$. In the following decoding step, the hidden state (memory) activation $\boldsymbol{s}$ of the RNN-based decoder at the l-th time step is computed as:

$$\boldsymbol{s}_l = \text{decoder}\left(\boldsymbol{s}_{l-1}, \boldsymbol{g}_l, \boldsymbol{y}_{l-1}\right), \quad (2)$$

where $\boldsymbol{g}_l$ and $\boldsymbol{y}_{l-1}$ denote the "glimpse" at the l-th time step and the predicted label at the previous step, respectively. Glimpse $\boldsymbol{g}_l$ is the weighted sum of the encoder output sequence:

$$\boldsymbol{g}_l = \sum_t \alpha_{l,t} \boldsymbol{h}_t, \quad (3)$$

where $\alpha_{l,t}$ is the attention weight of $\boldsymbol{h}_t$. It is calculated as:

$$\alpha_{l,t} = \exp(e_{l,t}) / \sum_{t'=1}^{T} \exp(e_{l,t'}), \quad (4)$$

$$e_{l,t} = \boldsymbol{w}^T tanh(\boldsymbol{W}\boldsymbol{s}_{l-1} + \boldsymbol{V}\boldsymbol{h}_t + \boldsymbol{U}f_{l,t} + \boldsymbol{b}), \quad (5)$$

$$\boldsymbol{f}_l = \boldsymbol{F} * \boldsymbol{\alpha}_{l-1}, \quad (6)$$

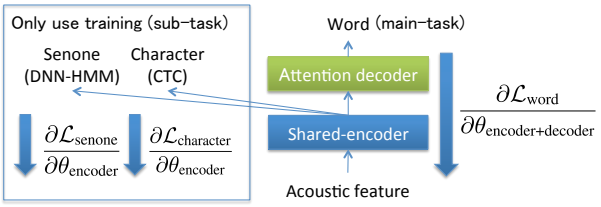


Figure 1: *The overall architecture for MT in this study. Each MT loss $\mathcal{L}$ of senone DNN-HMM-, character CTC- and word attention-based models is processed by Algorithm 1 and 2 for updating model parameter θ.*

where $\boldsymbol{U}$, $\boldsymbol{V}$, and $\boldsymbol{W}$ represent parameter matrices, and $\boldsymbol{b}$ is a bias vector. $*$ denotes 1-dimensional convolution and $\boldsymbol{F}$ is its parameter. Using $\boldsymbol{g}_l$ and $\boldsymbol{s}_{l-1}$, the decoder predicts the next label $\boldsymbol{y}_l$; we implement this as:

$$\boldsymbol{y}_l = \boldsymbol{R} \tanh\left(\boldsymbol{P}\boldsymbol{s}_{l-1} + \boldsymbol{Q}\boldsymbol{g}_l\right), \quad (7)$$

where $\boldsymbol{P}$, $\boldsymbol{Q}$, and $\boldsymbol{R}$ are parameter matrices.

The objective function for training the attention models is optimized by the cross entropy loss calculated between the predicted- and the target correct-sequences. For the acoustic-to-word speech recognition model with attention, we use two special labels representing the start- and end-of-sentence. The decoder completes decoding an utterance when the end-of-sentence is emitted. It is possible to conduct beam search to further enhance the recognition performance.

3. Multi-task learning (MT)

Fig. 1 illustrates the overall architecture of this study. Our architecture has one shared-encoder and three separate decoders. The decoders output senone, character or word. In this section, we first describe the often used conventional MT with IP, and then explain MT with AS as our proposal.

3.1. MT with interpolation (IP)

Algorithm 1 MT with interpolation (IP)

1: **for** all minibatches in training data **do**
2: pick minibatch i;
3: **for** all tasks in pool of tasks **do**
4: select task j to compute loss $\mathcal{L}_j$ for minibatch i;
5: **end for**
6: combine loss $\mathcal{L}$ from all tasks with preassigned weights λ_j for each task loss $\mathcal{L}_j$;
7: update neural network model with minibatch i;
8: **end for**

Often used MT with IP, called MT+IP, train the shared-encoder by individual task feedback, as is described in Algorithm 1. The objective function for MT is defined as the weighted sum of the losses $\mathcal{L}$ propagated from all branches; it is described as follows:

$$\mathcal{L} = \sum_j \lambda_j \mathcal{L}_j, \quad (8)$$

where $\mathcal{L}_j$ is the loss of the j-th task, and $\lambda_j \in [0, 1]$ is its interpolation weight. However, this approach does not explicitly learn each task loss due to averaging it. We want to explicitly use individual loss for more effective parameter updating.

3.2. MT with augmentation strategy (AS)

In this paper, we propose a framework complemented with MT and AS, called MT+AS. Instead of augmenting the data by

Algorithm 2 MT with augmentation strategy (AS)

```
decide main-task from pool of tasks;
for all minibatches in training data do
    pick minibatch i;
    for all sub-tasks in pool of tasks do
        select sub-task j to compute loss L_j for minibatch i;
        update neural network model with minibatch i;
    end for
    select main-task to compute loss L for minibatch i;
    update neural network model with minibatch i;
end for
```

applying various kinds of signal distortions to the input acoustic features as is often done, we augment the training data by creating multiple copies of the original data that suit the targets of each task. Also, our proposal sets the shared-encoder update order, as is illustrated in Algorithm 2. First, we select one main-task for the whole parameter $\theta_{\text{encoder+decoder}}$ updates. In our case, the main task consists of the encoder-decoder for word prediction. The sub-tasks consist then of the remaining tasks, here DNN-HMM senone prediction and CTC-based character prediction. For each mini-batch, we compute sequentially for each sub-task j the loss associated with that task, $\mathcal{L}_j$, and then update the parameters of the model associated with that sub-task using the error back-propagation algorithm. With this process, the shared-encoder parameter θ_{encoder} are gradually updated by each sub-task per minibatch. After performing the above process for each sub-task, the same process is finally performed for the main task. This ensures that at the end of the training the decoder associated with the main task is optimal for that task. We expect that our proposed training strategy will let the shared encoder learn a better intermediate representation that learns features common for all tasks, and that a word decoder can benefit from such a better encoder.

4. Experiments

4.1. Data

We evaluated our proposal using three actual Japanese speech recognition tasks: call center dialogue, and voice search task in just clean condition and in multi-conditions of Clean, Living room (TV chatting noise), and Kitchen (cooking and washing noise). The multi-conditions also include far-field scenario; the distance between the target speaker to the microphone array varies between 1 to 5 meters. The data amounts were 44, 112, and 1900 hours for training, and 7, 2, and 12 hours for evaluation. The evaluation set of voice search in multi-conditions is split to Clean, Living room, and Kitchen tasks. For the voice search task, the evaluation set covered the multi-conditions of Clean, Living room (TV noise and chatting), and Kitchen (washing and cooking noise), and balanced data wherein noisy conditions of Living room and Kitchen were extended with Clean. The number of symbols corresponding to the output units were 1568, 780, and 1142 targets for characters, and 3762, 1386, and 4911 targets for words. The word vocabulary holds words that appear more than three times. Others are treated as OOV words (UNK). The OOV rates in test sets of call center dialogue, voice search on clean and multi-conditions were 1.66%, 1.37% and 0.35%, respectively. We used label smoothing for improving generalization performance as described in [26]. The number of senones which are generated by force alignment with a GMM-HMM system, 3072, is common in all tasks.

Table 1: *WERs [%] of Japanese call center dialogue task (44 hours). "Use Labels" means the label used for MT. Each (·) in the MT+IP rows indicates the interpolation weight that yielded the best result in our experiment, and ⊙ in the MT+AS rows represents the update order of tasks.*

Approach	Use Labels			WER
	senone	character	word	
DNN-HMM	✓	-	-	27.41
ST	-	-	✓	30.41
ST + DAx2	-	-	✓	30.48
ST + DAx3	-	-	✓	30.23
ST + PT (senone)	-	-	✓	28.76
ST + PT (CTC)	-	-	✓	29.02
MT + IP	✓(0.5)	-	✓(0.5)	27.33
MT + IP	✓(0.2)	-	✓(0.8)	28.18
MT + IP	-	✓(0.5)	✓(0.5)	28.12
MT + IP	-	✓(0.2)	✓(0.8)	27.67
MT + IP	✓(0.3)	✓(0.3)	✓(0.4)	27.66
MT + IP (baseline)	✓(0.2)	✓(0.2)	✓(0.6)	27.14
MT + AS	✓①	-	✓②	26.27
MT + AS	-	✓①	✓②	25.29
MT + AS (best)	✓②	✓①	✓③	**24.39**
MT + AS	✓①	✓②	✓③	24.85

4.2. System configuration

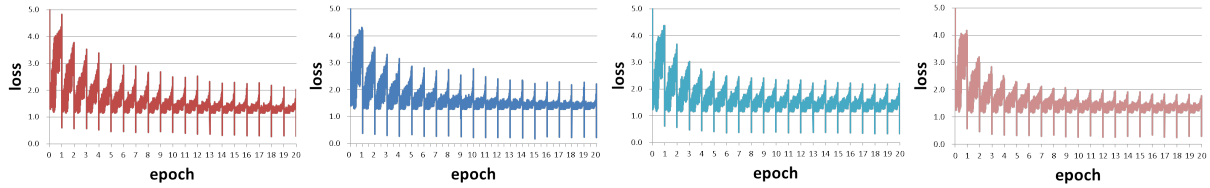
The input feature was 40 dimensional FBANK with non-overlapping frame stacking [8, 27]; three frames were stacked and skipped to make each new super-frame. The acoustic encoder in the attention-based model consists of five-layers of bidirectional LSTMs with 320 cells; the drop-out rate was 0.2 [28]. The attention-based word decoder consists of one-layer LSTM with 320 cells, a hidden layer with 320 tanh nodes, and a softmax output layer for word entries. The decoders of senone and CTC are only a softmax output layer for each entry. The weight of linear interpolation for MT+IP and the update order for MT+AS are described in the respective Tables.

We also built a DNN-HMM hybrid system using each dataset as baselines for comparison. The DNN-HMM system has six hidden layers with 2048 sigmoidal nodes and a softmax output layer with 3072 nodes. The language model was 3-gram and trained using a 1M Japanese web text corpus. For DNN-HMM system decoding, we used VoiceRex [29, 30].

We used the Adam optimizer with the setting described in [31]. We also used gradient clipping with a threshold of 5.0. The minibatch size was set to be 50 in all experiments. All network parameters were initialized with random values according to the setting in [32]. Since providing long input sequences can slow convergence at the beginning of the training, the input data were sorted by frame length before creating minibatches. We used the PyTorch toolkit to train the networks [33]. In decoding with the word-level attention model, we used simple beam search with beam width of 4. We evaluated performance in terms of word error rate (WER).

4.3. Results

Table 1 shows the ASR performance, i.e.WER, for the call center dialogue. "DNN-HMM" represents conventional DNN-HMM hybrid system as a refference. "ST" and "MT" indicate the use of single-task learning (only word attention-based model) and multi-task learning for training, respectively. "DA" means usual data augmentation with simply double or triple the amount of training data. This experiment is performed for a fair



(a) ST (b) ST + DAx3 (c) MT + IP (baseline) (d) MT + AS (best)

Figure 2: *The plot of training losses per epoch in the call center dialogue task. Each loss is normalized by the minibatch size. (a) - (d) represent the systems in Table 1. (a) single-task learning, (b) signle-task learning with data augmentation meaning triple the same training data, (c) conventional multi-task learning with interpolation, (d) multi-task learning with data augmentation by creating multiple copies with different labels that reflect corresponding task output targets. The end of each epoch yielded bad loss values because the input data were sorted by the length of frames before creating minibatches. Therefore, the bad losses in each epoch indicate long input sequences. Our proposal "MT+AS" yielded the fastest convergence of the long sequence losses.*

Table 2: *WERs [%] of Japanese voice search task in clean environment (112 hours). "Use Labels" means label used for MT. (·) in the MT+IP rows indicates the interpolation weight that yielded the best results in our experiment, and each ⊙ in MT+AS rows represents the update order of tasks.*

Approach	Use Labels			WER
	senone	character	word	
DNN-HMM	✓	-	-	9.78
ST	-	-	✓	10.07
ST + DAx2	-	-	✓	10.07
ST + DAx3	-	-	✓	10.18
ST + PT (senone)	-	-	✓	9.58
ST + PT (CTC)	-	-	✓	9.67
MT + IP	✓(0.5)	-	✓(0.5)	9.77
MT + IP (baseline)	✓(0.2)	-	✓(0.8)	9.68
MT + IP	-	✓(0.5)	✓(0.5)	10.06
MT + IP	-	✓(0.2)	✓(0.8)	9.89
MT + IP	✓(0.3)	✓(0.3)	✓(0.4)	9.71
MT + IP	✓(0.2)	✓(0.2)	✓(0.6)	9.83
MT + AS	✓①	-	✓②	9.20
MT + AS	-	✓①	✓②	9.37
MT + AS	✓②	✓①	✓③	9.19
MT + AS (best)	✓①	✓②	✓③	**9.03**

comparison with our proposal, which also augment the amount of training data in a similar way. "PT" uses pre-trained shared-encoder parameters with senone or character as the initialized parameters. "IP" and "AS" represent multi-task learning strategies (see in Section 3). We regard "MT+IP" as a baseline in this paper. We can see that our proposal, "MT+AS", is superior to the other systems with a significant WER improvement. Both MT+IP and MT+AS systems achieved the best WERs when using all target labels. The WER between MT+AS and simple ST is drastically improved, by up to 19.80% relative error reduction. The relative error reduction of our proposal over the best baseline is 8.14%. Moreover MT+AS is lower WER than DNN-HMM in spite of the small amount of training data. MT+IP is relatively sensitive to the choice of the interpolation weights. MT+IP could potentially achieve lower WER with further tuning, however there are infinite combination of weights possible, which makes such tuning hard. In contrast, our proposed method only needs to tune the order of the optimization of the sub-task, which requires only $N_{\text{sub-task}}!$ trials.

Table 2 shows the WERs for the voice search task in clean environment. We can also see that the MT+AS with all target labels achieved the best WER. The relative error reduction of our proposal over the best baseline using senone and word labels, is 6.71%. Here again, all MT+AS and majority of MT+IP approaches are suprerior to DNN-HMM.

Table 3: *WERs [%] of Japanese voice search task in multi-conditions of Clean environment, Living room and Kitchen as the noisy environments (1900 hours). MT+IP and MT+AS use all target labels.*

Approach	Clean	Living	Kitchen	AVG
DNN-HMM	10.06	10.19	9.05	9.66
ST	9.41	10.03	8.80	9.20
MT + IP	8.83	9.40	8.13	8.74
MT + AS	**8.05**	**8.92**	**7.78**	**8.10**

Table 3 lists the WERs for the voice search task in multi-conditions of Clean, Living room, and Kitchen. MT+IP and MT+AS use all target labels. The interpolation weights for MT+IP are senone(0.2), character(0.2) and word(0.6). The update order of MT+AS is first senone, second character, and finally word model parameters. This result also shows that our approach could attain the best WERs in all environments. The relative error reduction of our proposal over the best baseline is 7.32%. Moreover all attention-based models are superior to the DNN-HMM system on a large resource.

Finally, we analyze our proposal MT+AS effectiveness using the curve of training loss per epoch on call center dialogue task illustrated in Figure 2. The end of each epoch corresponds to the longest sentences of the training set as the sentences were sorted by length. We can see significantly faster for long sequences than the baseline systems. The number of the shared-encoder updates in MT+AS is more than ST and MT+IP. However ST+DAx3 did not converge faster than ST for long sequence although ST+DAx3 uses the same amount of training data than our proposed MT+AS. We assume that the ensemble feedbacks of each sub-task make the shared-encoder able to extract better intermediate representation, which helps training the word-based encoder-decoder model.

5. Conclusions

We have proposed multi-task learning with augmentation strategy that avoids being trapped in local optima and uses far fewer hyperparameters than linear interpolation. Our proposal performs multi-task learning with determined update order; the model is updated using main-task loss after updating by sub-task losses. We showed the effectiveness of the proposal in various Japanese speech recognition tasks where it achieved the best WERs. Comparisons with existing multi-task learning methods demonstrated that our proposal is very effective for all tasks, and offers the fastest convergence.

6. 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] J. J. Godfrey *et al.*, "Switchboard: Telephone speech corpus for research and development," in *ICASSP*, 1992, pp. 517–520.

[3] F. Jelinek, "Continuous speech recognition by statistical methods," in *Proc. IEEE*, vol. 64(4), 1976, pp. 532–556.

[4] G. Saon, G. Kurata, T. Sercu, K. Audhkhasi, S. Thomas, D. Dimitriadis, X. Cui, B. Ramabhadran, M. Picheny, L.-l. Lim, B. Roomi, and P. Hall, "English Conversational Telephone Speech Recognition by Humans and Machines," in *Proc. of INTERSPEECH*, 2017.

[5] W. Xiong, J. Droppo, X. Huang, F. Seide, M. Seltzer, A. Stolcke, D. Yu, and G. Zweig, "The Microsoft 2016 conversational speech recognition system," in *Proc. of ICASSP*, 2017, pp. 5255–5259.

[6] A. Graves, S. Fernandez, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification : Labelling unsegmented sequence data with recurrent neural networks," in *Proc. of ICML*, 2006, pp. 369–376.

[7] A. Graves and N. Jaitly, "Towards End-To-End speech recognition with recurrent neural networks," in *Proc. of ICLR*, 2014, pp. 1764–1772.

[8] H. Sak, A. Senior, K. Rao, and F. Beaufays, "Fast and accurate recurrent neural network acoustic models for speech recognition," *arXiv preprint arXiv:1507.06947*, 2015.

[9] H. Sak, F. de Chaumont Quitry, T. Sainath, and K. Rao, "Acoustic modelling with CD-CTC-SMBR LSTM RNNs," in *Proc. of ASRU*. IEEE, 2015, pp. 604–609.

[10] J. Chorowski, D. Bahdanau, K. Cho, and Y. Bengio, "End-to-end continuous speech recognition using attention-based recurrent NN: first results," in *Proc. of NIPS*, 2014.

[11] J. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio, "Attention-based models for speech recognition," in *Proc. of NIPS*, 2015, pp. 577–585.

[12] R. Prabhavalkar, T. N. Sainath, B. Li, K. Rao, and N. Jaitly, "An analysis of "attention" in sequence-to-sequence models," in *Proc. of INTERSPEECH*, 2017, pp. 3702–3706.

[13] W. Chan, N. Jaitly, Q. Le, and O. Vinyals, "Listen, attend and spell: A neural network for large vocabulary conversational speech recognition," in *Proc. of ICASSP*, 2016, pp. 4960–4964.

[14] D. Bahdanau, J. Chorowski, D. Serdyuk, P. Brakel, and Y. Bengio, "End-to-End attention-based large vocabulary speech recognition," in *Proc. of ICASSP*, 2016, pp. 4945–4949.

[15] Y. Miao, M. Gowayyed, and F. Metze, "EESEN: End-to-end speech recognition using deep RNN models and WFST-based decoding," in *Proc. of ASRU*, 2015, pp. 167–174.

[16] K. Audhkhasi, B. Ramabhadran, G. Saon, M. Picheny, and D. Nahamoo, "Direct acoustics-to-word models for English conversational speech recognition," in *Proc. of INTERSPEECH*, 2017, pp. 959–963.

[17] H. Soltau, H. Liao, and H. Sak, "Neural speech recognizer: Acoustic-to-word LSTM model for large vocabulary speech recognition," in *Proc. of INTERSPEECH*, 2017, pp. 3707–3711.

[18] L. Lu, X. Zhang, and S. Renals, "On training the recurrent neural network encoder-decoder for large vocabulary end-to-end speech recognition," in *Proc. of ICASSP*, 2016, pp. 5060–5064.

[19] T. Hori, S. Watanabe, Y. Zhang, and W. Chan, "Advances in joint CTC-attention based end-to-end speech recognition with a deep CNN encoder and RNN-LM," in *Proc. of INTERSPEECH*, 2017, pp. 949–953.

[20] T. Hori, S. Watanabe, and J. R. Hershey, "Multi-level language modeling and decoding for open vocabulary end-to-end speech recognition," in *Proc. of ASRU*, 2017, pp. 287–293.

[21] S. Ueno, H. Inaguma, M. Mimura, and T. Kawahara, "Acoustic-to-word attention-based model complemented with character-level CTC-based model," in *Proc. of ICASSP*, 2018, pp. 5804–5808.

[22] S. Toshniwal, H. Tang, L. Lu, and K. Livescu, "Multitask learning with low-level auxiliary tasks for encoder-decoder based speech recognition," in *Proc. of INTERSPEECH*, 2017, pp. 3532–3536.

[23] S. Kim, T. Hori, and S. Watanabe, "Joint CTC-attention based end-to-end speech recognition using multi-task learning," in *Proc. of ICASSP*, 2017, pp. 4835–4839.

[24] K. Rao, H. Sak, and R. Prabhavalkar, "Exploring architectures, data and units for streaming end-to-end speech recognition with rnn-transducer," in *Proc. of ASRU*, 2017, pp. 206–213.

[25] T. Fukuda, M. Suzuki, G. Kurata, S. Thomas, J. Cui, and B. Ramabhadran, "Efficient knowledge distillation from an ensemble of teachers," in *Proc. of INTERSPEECH*, 2017, pp. 3697–3701.

[26] G. Pereyra, G. Tucker, J. Chorowski, L. Kaiser, and G. E. Hinton, "Regularizing neural networks by penalizing confident output distributions," vol. abs/1701.06548, 2017.

[27] G. Pundak and T. N. Sainath, "Lower frame rate neural network acoustic models," in *Proc. of INTERSPEECH*, 2016, pp. 22–26.

[28] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, pp. 1929–1958, 2014.

[29] 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.

[30] T. Hori, C. Hori, Y. Minami, and A. 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.

[31] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, pp. 1–15, 2014.

[32] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *Proc. of ICCV*, 2015, pp. 1026–1034.

[33] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in PyTorch," in *NIPS-W*, 2017.