SEQUENCE-LEVEL KNOWLEDGE DISTILLATION FOR MODEL COMPRESSION OF ATTENTION-BASED SEQUENCE-TO-SEQUENCE SPEECH RECOGNITION

Raden Mu'az Mun'im, Nakamasa Inoue, Koichi Shinoda

Tokyo Institute of Technology
raden@ks.c.titech.ac.jp, inoue@ks.c.titech.ac.jp, shinoda@c.titech.ac.jp

ABSTRACT

We investigate the feasibility of sequence-level knowledge distillation of Sequence-to-Sequence (Seq2Seq) models for Large Vocabulary Continuous Speech Recognition (LVCSR). We first use a pre-trained larger teacher model to generate multiple hypotheses per utterance with beam search. With the same input, we then train the student model using these hypotheses generated from the teacher as pseudo labels in place of the original ground truth labels. We evaluate our proposed method using Wall Street Journal (WSJ) corpus. It achieved up to $9.8\times$ parameter reduction with accuracy loss of up to 7.0% word-error rate (WER) increase.

Index Terms— speech recognition, large vocabulary continuous speech recognition, sequence-to-sequence, attention model, knowledge distillation, sequence-level knowledge distillation

1. INTRODUCTION

In recent years, end-to-end deep neural networks for Large Vocabulary Continuous Speech Recognition (LVSCR) have been steadily improving their accuracy, rivalling the traditional Gaussian Mixture Model-Hidden Markov Models (GMM-HMM) and hybrid models of deep neural networks and HMMs (DNN-HMM). While these models have an acoustic model and a language model which are trained separately, for end-to-end training, the whole model is trained using backpropagation with audio-transcription pairs [1].

Several end-to-end architectures for speech recognition have been proposed. Their examples include Connectionist Temporal Classification (CTC) [2], Recurrent Neural Network-Transducer (RNN-T) [3], and Sequence-to-Sequence (Seq2Seq) with attention [4]. Unlike CTC and RNN-T, Seq2Seq with attention does not make any prior assumptions on the output sequence alignment given an input; it jointly learns how to align while learning to encode the input and decode its result into the output.

In machine learning, model compression [5] is a way to significantly compress a model by reducing the number of its parameters while having negligible accuracy loss. This is important for deploying trained models on memory and compute-constrained devices such as mobile phones and embedded systems, and when energy efficiency is needed on large-scale deployment. Several model compression methods exist, such as pruning, quantization [5], and knowledge distillation (KD) [6, 7]. KD is the focus of this work, where a smaller student model is trained by using the output distribution of a larger teacher model. Recently KD for Seq2Seq models with attention was proposed for neural machine translation (NMT) task [8] and proved to be effective. Also, Seq2Seq models for CTC-based speech recognition was recently proposed [9].

In this paper, we propose a sequence-level KD method for Seq2Seq speech recognition models with attention. Different from the previous work for NMT [8], we extract the hypotheses from a pre-trained teacher Seq2Seq model using beam search, and train student Seq2Seq models using the hypotheses as pseudo labels on the sequence-level cross-entropy criterion. To the best of our knowledge, there is no prior work on applying KD in Seq2Seq-based speech recognition models.

2. KNOWLEDGE DISTILLATION FOR SEQ2SEQ MODELS

2.1. Sequence-to-Sequence Learning

The Sequence-to-Sequence (Seq2Seq) [1] is neural network architecture which directly models conditional probability $p(\mathbf{y}|\mathbf{x})$ where $\mathbf{x} = [x_1, ..., x_S]$ is the source sequence with length S and $\mathbf{y} = [y_1, ..., y_T]$ is the target sequence with length T.

Figure 1 illustrates the Seq2Seq model. It consists of an encoder, a decoder and an attention module. The encoder processes an input sequence $\mathbf{x}$ and outputs encoded hidden representation $\mathbf{h^e} = [h_1^e, ..., h_S^e]$ for the decoder [10]. The attention module is a network that assists the decoder to find relevant information on the encoder side based on the current decoder hidden states [4]. The attention module does this by producing a context vector c_t at time t based on the encoder

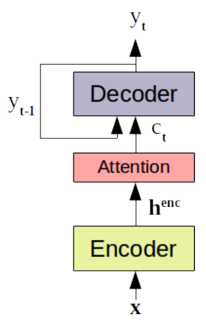

Fig. 1. Architecture of Seq2Seq with Attention module

and decoder hidden states:

$$c_t = \sum_{s=1}^{S} a_t(s) * h_s^e, \tag{1}$$

$$a_t(s) = \frac{\exp(\text{Score}(h_s^e, h_t^d))}{\sum_{s=1}^{S} \exp(\text{Score}(h_s^e, h_t^d))}, \tag{2}$$

Several variations exist for $\text{Score}(h_s^e, h_t^d)$:

$$\text{Score}(h_s^e, h_t^d) = \begin{cases} \langle h_s^e, h_t^d \rangle, & : \text{dot product} \\ h_s^{e\intercal} W_s h_t^d, & : \text{bilinear} \\ V_s^\intercal \tanh(W_s[h_s^e, h_t^d]), & : \text{MLP} \end{cases} \tag{3}$$

where Score is a function $(\mathbb{R}^M \times \mathbb{R}^N) \rightarrow \mathbb{R}$, M is the number of hidden units for the encoder, N is the number of hidden units for the decoder, and both W_s and $V_s^\intercal$ are weight matrices. Finally, the decoder predicts the probability of target sequence $\mathbf{y}$ at time t based on the previous output $y_{<t}$ and c_t, which can be formulated as:

$$\log p(\mathbf{y}|\mathbf{x}) = \sum_{t=1}^{T} \log p(y_t|y_{<t}, c_t). \tag{4}$$

The previous outputs c_t can obtained with greedy decoding, i.e. by taking the output with the highest probability for each time step (e.g in [8]). It is also possible to perform *beam search* to obtain more than one c_t [1].

Seq2Seq can handle virtually any sequence related tasks[1], such as NMT and speech recognition. For speech recognition, the input $\mathbf{x}$ is a sequence of feature vectors derived from audio waveform such as Short-time Fourier Transform (STFT) spectrogram or Mel-frequency cepstral coefficients (MFCC). Therefore, $\mathbf{x}$ is a real vector with $S \times D$ dimension where D is the number of the features and S is the total length of the utterance in frames. The output $\mathbf{y}$ can be a sequence of phonemes or graphemes (characters).

2.2. Knowledge Distillation

In knowledge distillation (KD), a teacher model's probability distribution $q(\theta_T|x;\theta)$ is trained by using a given dataset, where θ_t is a set of its parameters. Using the same dataset, a student model $p(\theta|x;\theta)$ is trained by minimizing cross-entropy with the teacher model's probability distribution $q(\theta_T|x;\theta)$ instead of the ground truth labels from the dataset.

Let an input-label pairs be (x, y) and $\mathcal{V}$ is a set of possible classes. Then, the loss for KD is given as:

$$\mathcal{L}_{\text{KD}}(\theta;\theta_T) = -\sum_{k=1}^{|\mathcal{V}|} q(y=k\,|\,x;\theta_T)\log p(y=k\,|\,x;\theta). \tag{5}$$

In KD training, the teacher produces softmax probabilities (*soft targets*), which reveal teacher's confidences for what classes it predicts given an input. With this additional knowledge, the student can model the data distribution better than learning directly from the ground truth labels consisting of one-hot vectors (*hard targets*) [7].

2.3. Sequence-Level Knowledge Distillation

While it is possible to use the original KD method to train autoregressive models such as RNN, it only gives non-significant accuracy gains [8], or simply degrade the performance compared to training with the dataset directly [11].

To adapt KD to autoregressive models, [8] suggested to use the approximation of a teacher's sequence-level distribution instead of the teacher's single time step frame-level distribution, so as to capture the time-dependencies between the inputs.

Consider the sequence-level distribution specified by the model over all possible teacher label sequences $\mathbf{t} \in \mathcal{T}$, given the input sequence $\mathbf{s}$:

$$p(\mathbf{t}\,|\,\mathbf{s}) = \prod_{j=1}^{J} p(t_j\,|\,\mathbf{s}, \mathbf{t}_{<j}), \tag{6}$$

for any length J. The sequence-level loss for Seq2Seq involves matching the input $\mathbf{s}$ with the one-hot distribution of all the complete sequences:

$$\mathcal{L}_{\text{SEQ-NLL}} = -\sum_{\mathbf{t}\in\mathcal{T}} \mathbb{1}\{\mathbf{t}=\mathbf{y}\}\log p(\mathbf{t}\,|\,\mathbf{s}) \tag{7}$$

$$= \quad -\sum_{j=1}^{J}\sum_{k=1}^{|\mathcal{V}|} \mathbb{1}\{y_j=k\}\log p(t_j=k\,|\,\mathbf{s},\mathbf{t}_{<j}), \tag{8}$$

where $\mathbb{1}\{\cdot\}$ is the indicator function and $\mathbf{y} = [y_1, \ldots, y_J]$ is the observed sequence.

To formulate the sequence-level KD, we use $q(\mathbf{t}\,|\,\mathbf{s})$ to represent the teacher's sequence distribution over the sample

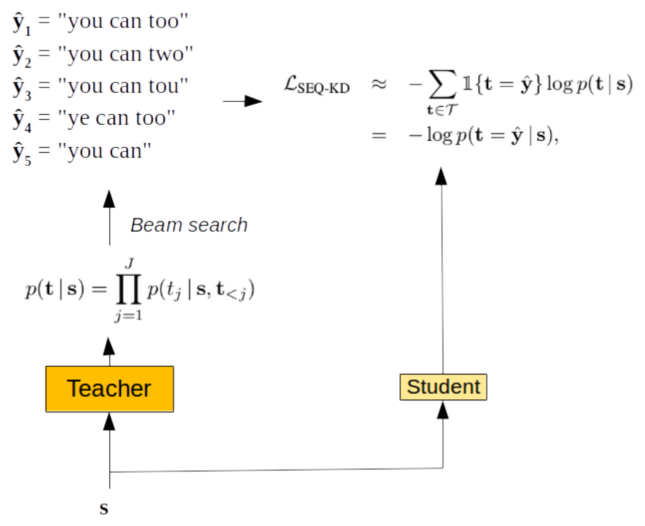
$\hat{\mathbf{y}}_1$ = "you can too"
$\hat{\mathbf{y}}_2$ = "you can two"
$\hat{\mathbf{y}}_3$ = "you can tou"
$\hat{\mathbf{y}}_4$ = "ye can too"
$\hat{\mathbf{y}}_5$ = "you can"
$\mathcal{L}_{\text{SEQ-KD}} \approx -\sum_{\mathbf{t}\in\mathcal{T}} \mathbb{1}\{\mathbf{t}=\hat{\mathbf{y}}\}\log p(\mathbf{t}\,|\,\mathbf{s}) = -\log p(\mathbf{t}=\hat{\mathbf{y}}\,|\,\mathbf{s}),$
Beam search
$p(\mathbf{t}\,|\,\mathbf{s}) = \prod_{j=1}^{J} p(t_j\,|\,\mathbf{s},\mathbf{t}_{<j})$
Teacher
Student
s

Fig. 2. Example of sequence-level KD training. The teacher produces pseudo labels obtained from the top-k results from beam search, then the student learns by minimizing cross entropy with them.

space of all possible sequences,

$$\mathcal{L}_{\text{SEQ-KD}} = -\sum_{\mathbf{t}\in\mathcal{T}} q(\mathbf{t}\,|\,\mathbf{s})\log p(\mathbf{t}\,|\,\mathbf{s}) \tag{9}$$

Different from previously stated $\mathcal{L}_{\text{KD}}$, $\mathcal{L}_{\text{SEQ-KD}}$ minimizes the loss on the whole-sequence level. However, this loss is intractable. An approximation to calculate it is required.

There are many ways to approximate the loss. The sequence-level KD for NMT [8] uses a single hypothesis with the best score as the pseudo label per input. For CTC-based speech recognition [9], this loss was k-best hypotheses from beam search per input (Figure 2). The loss is then

$$\begin{aligned}\mathcal{L}_{\text{SEQ-KD}} &\approx -\sum_{\mathbf{t}\in\mathcal{T}} \mathbb{1}\{\mathbf{t}=\hat{\mathbf{y}}\}\log p(\mathbf{t}\,|\,\mathbf{s}) &(10)\\ &= -\log p(\mathbf{t}=\hat{\mathbf{y}}\,|\,\mathbf{s}), &(11)\end{aligned}$$

where $\hat{\mathbf{y}}$ is the output hypothesis obtained from running beam search with the teacher model. In this work, we investigated on how to apply these for Seq2Seq-based speech recognition models, which will be discussed in the next sections.

3. MULTIPLE HYPOTHESES FROM BEAM SEARCH FOR KNOWLEDGE DISTILLATION

For Seq2Seq-based speech recognition, the sequence-level KD with the loss approximation can be done by the following three steps: (1) Pre-train a teacher model with a dataset, (2) With the same dataset, generate k-best sequences with beam search, and save them as pseudo labels, (3) Train a student model with the same dataset but replace the ground truth labels with generated pseudo labels (Figure 2). In this procedure, the size of beam search and the value of k are adjustable hyperparameters. The dataset size increases with factor of k from the method using the 1-best [8].

The pseudo labels are analogous to *soft targets* in the original KD. Even if the pseudo labels are not fully accurate, the student is expected to achieve better performance with this training method since the student tries to imitate the teacher's distribution instead of trying to model the training data distribution directly. Training with these pseudo labels can be seen as a form of regularization similar to the original KD [7], because the slightly inaccurate transcriptions from pseudo labels can prevent the trained models from overfitting to training data distribution (Figure 2).

Model	Encoder bi-GRU	Decoder GRU
Teacher	5 layers 384 cells	3 layers 384 cells
Student-mid	4 layers 256 cells	1 layer 256 cells
Student-small	3 layers 128 cells	1 layer 128 cells

Table 1. Model configurations. 2D CNN and Attention module configuration is the same for all models.

4. IMPLEMENTATION AND EXPERIMENTS

4.1. Seq2Seq Model Architecture

The input speech audio waveform is sampled at 16kHz, then transformed into STFT spectrogram with Hanning window of 20ms and step size of 10ms. Then the STFT spectrograms are fed into 2D convolutional neural network (CNN) with two layers, as described by [12, 13], as this STFT and CNN combination can further improve accuracy compared to using MFCC alone. 2D CNN is configured with filters=32, kernel_size=(5, 8), stride=(2,2).

Gated Recurrent Unit (GRU) [14] is used for the encoder and the decoder. The feature vector from 2D CNN is first fed into the encoder. Then, the hidden representation made by the encoder is fed to an embedding layer of size 32. Next, the output of the embedding layer is fed into the decoder.

The output of the decoder is fed into the softmax layer consisting of 31 classes (26 English alphabet plus 5 classes for start-of-sentence (sos), end-of-sentence (eos), space, apostrophe and period). The decoder's attention module consists of 1D CNN layer with kernel=128, kernel_size=15, stride=1, padding=7 followed by a fully-connected layer, as proposed by [4]. The models are trained with Dropout set to 0.4.

The model configurations are summarised in Table 1. We have designed two kinds of student models with different sizes, Student-mid and Student-small.

Table 2. Results on WSJ eval92 (trained on train_si284). beamSize (shorthand for size of beam search) and topK (shorthand for top *k*-best hypotheses) are hyperparameters. We measured the accuracy using the character-error rate (CER) and word-error rate (WER).

Model	CER (%)	WER (%)
Teacher (Params: 16.8M; 100% size)		
Baseline	4.6	15.3
Student-mid (Params: 6.1M; 37% size)		
Baseline	7.0	21.8
topK=1, beamSize=5	6.4	20.5
topK=1, beamSize=10	6.5	20.5
topK=5, beamSize=5	**6.0**	20.1
topK=5, beamSize=10	6.5	21.2
topK=10, beamSize=10	6.1	**19.7**
Student-small (Params: 1.7M; 10% size)		
Baseline	9.2	28.7
topK=1, beamSize=5	7.6	26.1
topK=1, beamSize=10	7.7	25.3
topK=5, beamSize=5	**6.5**	**22.3**
topK=5, beamSize=10	6.9	23.3
topK=10, beamSize=10	7.4	24.7

4.2. Dataset and Software

Wall Street Journal (WSJ) corpus (available at the Linguistic Data Consortium as LDC93S6B and LDC94S13B) is used as the training and testing datasets. From the corpus, train_si284 (81 hours of 37k sentences) is used for training, dev93 is used for validation, and eval92 is used for testing. Data is extracted as according to WSJ recipe from Kaldi toolkit [15]. STFT spectrogram extraction, implementation of the models, training and testing are conducted using Python 3.6, Scipy 1.0.1 and Pytorch 0.3.1.

4.3. Experimental Setup

First, the teacher model and student models are trained directly with train_si284 and tested with eval92 to serve as the baselines. To perform Sequence-level KD, the teacher model pre-trained with train_si284 is used for pseudo labels generation. This is done by extracting *k*-best hypotheses from beam search with combinations of beam size of 5, 10 and *k*-best of 1, 5, 10.

For each model the training is done using Adam optimizer, with learning rate of 2e-4 exponential decay rate of 0.99 per epoch, and mini-batch size set at 16. Seq2Seq teacher forcing rate [4] is set at 0.4. Training is done up to 200 epochs, or until no improvement can be seen in validation or testing set.

5. RESULTS

The results are shown in Table 2. The teacher model serves the reference baseline for student models. We benchmarked the character-error rate (CER) and word-error rate (WER). With KD training, the student models managed to achieve better CER and WER than the case when training directly with the dataset.

For Student-mid model, the number of parameters is 37% of the teacher (2.7× reduction). It achieved the best CER (6.0%) with beamSize=5 and topK=5, and the best WER (19.7%) with beamSize=10 and beamSize=10.

For Student-small model, the number of parameters is 10% of the teacher (9.8× reduction). As expected, the model suffered higher CER and WER compared to Student-mid model. The model achieved the best CER (6.5%) and WER (22.3%) with beamSize=5 and topK=5. The effect of sequence-level KD is more obvious in Student-small, where it achieve 6.4% reduction in WER with KD training compared to directly training with the dataset.

Training was done using a server with Intel Xeon E5-2680 2.4GHz and NVIDIA Tesla P100 16GB. We did not found significant improvement in training and testing time. Student-mid and Student-small achieved speed-up of 1.4× and 1.7× respectively, relative to the teacher model. This relatively small improvement may be due to inherent sequential computations using RNN where some operations simply cannot be paralellized.

To summarise, generally we found that using beamSize=5 and topK=5 are sufficient for reasonable WER and CER reduction. Increasing beamSize and/or topK further do not necessarily improve the performance.

6. CONCLUSION

In this work, we successfully performed sequence-level KD training for Seq2Seq speech recognition models. Using beam search to approximate the teacher's distribution, we extracted *k*-best hypotheses to be used as pseudo labels to train the student on the same dataset. We managed to train the students with reduction of 9.8× parameter size of the teacher, with increase of WER of 7.0% relative to the teacher.

There are many problems left for future work. Since RNN is limited in inference speed due to its sequential operations, we plan to investigate the feasibility of sequence-level KD for other highly parallelizable attention-based architectures which are not based on RNN, such as, Transformer [16] and S2SConv [17].

7. ACKNOWLEDGMENT

This work was supported by JST CREST Grant Number JP-MJCR1687 and JSPS KAKEN Grant Number 16H02845.

8. REFERENCES

[1] Ilya Sutskever, Oriol Vinyals, and Quoc V Le, "Sequence to sequence learning with neural networks," in *Advances in neural information processing systems*, 2014, pp. 3104–3112.

[2] Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton, "Speech recognition with deep recurrent neural networks," in *Acoustics, speech and signal processing (ICASSP), 2013 IEEE international conference on*. IEEE, 2013, pp. 6645–6649.

[3] Alex Graves and Navdeep Jaitly, "Towards end-to-end speech recognition with recurrent neural networks," in *International Conference on Machine Learning*, 2014, pp. 1764–1772.

[4] Dzmitry Bahdanau, Jan Chorowski, Dmitriy Serdyuk, Philemon Brakel, and Yoshua Bengio, "End-to-end attention-based large vocabulary speech recognition," in *Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on*. IEEE, 2016, pp. 4945–4949.

[5] Song Han, Huizi Mao, and William J Dally, "Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding," *arXiv preprint arXiv:1510.00149*, 2015.

[6] Jinyu Li, Rui Zhao, Jui-Ting Huang, and Yifan Gong, "Learning small-size dnn with output-distribution-based criteria," in *Interspeech 2014*, 2014.

[7] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean, "Distilling the knowledge in a neural network," *arXiv preprint arXiv:1503.02531*, 2015.

[8] Yoon Kim and Alexander M Rush, "Sequence-level knowledge distillation," in *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, 2016, pp. 1317–1327.

[9] Ryoichi Takashima, Sheng Li, and Hisashi Kawai, "An investigation of a knowledge distillation method for ctc acoustic models," in *Acoustics, Speech and Signal Processing (ICASSP), 2018 IEEE International Conference on*. IEEE, 2018, pp. 5809–5813.

[10] Andros Tjandra, Sakriani Sakti, and Satoshi Nakamura, "Attention-based wav2text with feature transfer learning," in *Automatic Speech Recognition and Understanding Workshop (ASRU), 2017 IEEE*. IEEE, 2017, pp. 309–315.

[11] Haşim Sak, Félix de Chaumont Quitry, Tara Sainath, Kanishka Rao, et al., "Acoustic modelling with cd-ctc-smbr lstm rnns," in *Automatic Speech Recognition and Understanding (ASRU), 2015 IEEE Workshop on*. IEEE, 2015, pp. 604–609.

[12] Andrew L Maas, Peng Qi, Ziang Xie, Awni Y Hannun, Christopher T Lengerich, Daniel Jurafsky, and Andrew Y Ng, "Building dnn acoustic models for large vocabulary speech recognition," *Computer Speech & Language*, vol. 41, pp. 195–213, 2017.

[13] Dario Amodei, Sundaram Ananthanarayanan, Rishita Anubhai, Jingliang Bai, Eric Battenberg, Carl Case, Jared Casper, Bryan Catanzaro, Qiang Cheng, Guoliang Chen, et al., "Deep speech 2: End-to-end speech recognition in english and mandarin," in *International Conference on Machine Learning*, 2016, pp. 173–182.

[14] Junyoung Chung, Caglar Gulcehre, Kyunghyun Cho, and Yoshua Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," in *NIPS 2014 Workshop on Deep Learning, December 2014*, 2014.

[15] Daniel Povey, Arnab Ghoshal, Gilles Boulianne, Lukas Burget, Ondrej Glembek, Nagendra Goel, Mirko Hannemann, Petr Motlicek, Yanmin Qian, Petr Schwarz, et al., "The kaldi speech recognition toolkit," in *IEEE 2011 workshop on automatic speech recognition and understanding*. IEEE Signal Processing Society, 2011, number EPFL-CONF-192584.

[16] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems*, 2017, pp. 5998–6008.

[17] Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N Dauphin, "Convolutional sequence to sequence learning," in *International Conference on Machine Learning*, 2017, pp. 1243–1252.