KNOWLEDGE DISTILLATION FOR NEURAL TRANSDUCERS FROM LARGE SELF-SUPERVISED PRE-TRAINED MODELS

Xiaoyu Yang, Qiujia Li[*]*, Philip C. Woodland*

Cambridge University Engineering Dept., Trumpington St., Cambridge, CB2 1PZ U.K.

{xy316,ql264,pcw}@eng.cam.ac.uk

ABSTRACT

Self-supervised pre-training is an effective approach to leveraging a large amount of unlabelled data to reduce word error rates (WERs) of automatic speech recognition (ASR) systems. Since it is impractical to use large pre-trained models for many real-world ASR applications, it is desirable to have a much smaller model while retaining the performance of the pre-trained model. In this paper, we propose a simple knowledge distillation (KD) loss function for neural transducers that focuses on the one-best path in the output probability lattice under both streaming and non-streaming setups, which allows a small student model to approach the performance of the large pre-trained teacher model. Experiments on the LibriSpeech dataset show that despite being 10 times smaller than the teacher model, the proposed loss results in relative WER reductions (WERRs) of 11.5% and 6.8% on the test-other set for non-streaming and streaming student models compared to the baseline transducers trained without KD using the labelled 100-hour clean data. With an additional 860 hours of unlabelled data for KD, the WERRs increase to 48.2% and 38.5% for non-streaming and streaming students. If language model shallow fusion is used for producing distillation targets, a further improvement in the student model is observed.

Index Terms— knowledge distillation, neural transducer, ASR

1. INTRODUCTION

Self-supervised learning has emerged as a paradigm to learn general data representations. The model is first pre-trained on a very large amount of unlabelled data and then fine-tuned with task-specific labelled data. The effectiveness of this approach was first demonstrated in various natural language processing tasks [1, 2] and, more recently, in automatic speech recognition (ASR) [3, 4]. Promisingly, some of these models could reach state-or-the-art performance without relying on a large amount of labelled data. Although self-supervised pre-training can give excellent performance for ASR, the models are typically very large with hundreds of millions or even billions of parameters. This leads to large memory requirements and a long inference time, which hinders the direct deployment of such models for real-world ASR applications [5]. Moreover, the pre-trained speech encoder is generally bidirectional or attends to a very wide range of past and future context [3, 4], which is problematic for ASR applications that require streaming processing of speech data.

To address this issue, various model compression methods have been proposed to reduce the model size dramatically with only a moderate increase in the WER. For example, singular value decomposition of weight matrices can reduce the number of parameters in acoustic models using feed-forward networks [6] and recurrent neural networks (RNNs) [7] and a sparse pruning method can shrink the model size by setting weights with small values to zero [8]. Instead of direct manipulation of model parameters, knowledge distillation (KD) [9] is an alternative approach for model compression, where a small student model learns to mimic the output behaviour of a large teacher model. As KD only depends on the outputs from the teacher and student models, the student model is free to have a different representation of the input data and also a different model architecture. KD is also able to take advantage of unlabelled data as only the output from the teacher model is needed to train the student model [10, 11]. For conventional hidden Markov model-based systems, KD has been applied to the acoustic model at the frame level [12] and the sequence level [13]. For end-to-end (E2E) ASR, KD has been applied for connectionist temporal classification (CTC) models [14–17] and attention-based encoder-decoder models [17, 18] by either matching the hidden representations or the softmax output distributions between the teacher and student models. The neural transducer [19] is another widely adopted E2E ASR model because of its competitive performance and readiness for streaming applications [20, 21]. However, applying KD to neural transducers is less straightforward as computing the distillation loss on the whole output probability lattice from transducers is very expensive and memory-inefficient, especially for long sequences. [11] trained the student transducer model directly from the "pseudo transcriptions", *i.e.* erroneous recognition results from the teacher model, instead of using the output distributions. [22] carried out KD for transducers using a collapsed version of the full output distributions to reduce the memory requirement during training. As a workaround, [23] performs KD over the encoder hidden representations.

In this paper, we investigate how to effectively distil knowledge from a large pre-trained transducer model to a much smaller one with a limited amount of labelled data. Instead of using the collapsed distribution over the whole probability lattice [22], using the full distribution of the one-best path in the output probability lattice yields better performance. The proposed distillation loss can be simply adapted to transfer knowledge from a non-streaming teacher to a streaming student by introducing a time-shift variable that controls the amount of future context seen by the encoder. In our experiments, the teacher is a wav2vec 2.0 model [4] with the waveform as input, whereas the student is a Conformer model [24] with filter bank features as input. Although the student model is 10 times smaller than the teacher model, experiments on LibriSpeech show that the proposed method significantly improves the student's performance for both non-streaming and streaming cases. Further word error rate (WER) reduction is obtained by using unlabelled data for KD or augmenting the distillation targets by a language model (LM).

In the rest of this paper, Sec. 2 briefly reviews self-supervised pre-training for ASR and the neural transducer models. In Sec. 3, the proposed distillation loss for streaming and non-streaming transducers are introduced. The experimental setup and results are described in Secs. 4 and 5. Conclusions are drawn in Sec. 6.

[*]Li is funded by a Peterhouse Graduate Studentship.

2. SELF-SUPERVISED PRE-TRAINING FOR E2E ASR

2.1. wav2vec 2.0 for Self-Supervised Pre-training

Self-supervised learning for E2E ASR has shown promising results in the last few years [3, 4, 25]. Among these methods, wav2vec 2.0 (w2v2) is one of the most effective frameworks. The raw speech waveform $\mathbf{w}$ is first processed by a convolutional neural network feature extractor $\mathcal{M}$ to generate latent speech representations $\mathcal{M}(\mathbf{w}) = \mathbf{z}_{1:T}$, which are then sent to a Transformer network $\mathcal{N}$ to generate contextualised speech representations $\mathcal{N}(\mathbf{z}_{1:T}) = \mathbf{c}_{1:T}$. A quantisation module $\mathcal{Q}$ discretises the latent speech representation to $\mathbf{q}_t$. Pre-training is carried out by predicting the correct quantised representations of the masked latent speech representations from a set of distractors. This training objective for each $\mathbf{z}_t$ can be expressed with a contrastive loss function Eqn. (1), where Q_t is a set consisting of the correct quantisation $\mathbf{q}_t$ and other distractors, κ is the temperature and SIM denotes a similarity measure. An additional quantisation diversity loss is added to construct the final loss during pre-training.

$$\mathcal{L}_{\text{contrastive}} = -\log \frac{\exp(\text{SIM}(\mathbf{c}_t, \mathbf{q}_t)/\kappa)}{\sum_{\mathbf{q}\in Q_t} \exp(\text{SIM}(\mathbf{c}_t, \mathbf{q})/\kappa)}. \quad (1)$$

The pre-trained model is used to initialise the encoder of an E2E ASR model. The encoder is then fine-tuned together with additional output layers or decoders using labelled data for ASR. Although the original pre-trained models are fine-tuned with the CTC objective [4], they can also be used to initialise transducer models [26].

2.2. Neural Transducer Model

CTC [27] is one of the earliest frameworks for E2E ASR. Since the CTC assumption that output tokens at different timestamps are conditionally independent is not realistic for ASR, the RNN transducer was proposed to address this limitation [19]. More recently, RNNs in the transducer encoder have been replaced in order to give improved performance and leading to Transformer transducers [28] and Conformer transducers [24]. A neural transducer model consists of an encoder $\mathcal{F}$, a prediction network $\mathcal{G}$ and a joint network $\mathcal{J}$. During training, given an input feature sequence $\mathbf{X} = \mathbf{x}_{1:T}$ of length T, the encoder $\mathcal{F}$ extracts the latent representations $\mathbf{f}_{1:T}$. The prediction network predicts the next symbol y_u given all previous symbols $y_{1:u-1}$ for $u = 1, \ldots, U$, similar to a language model. The output from the prediction network $\mathcal{G}(\mathbf{y}) = \mathbf{g}_{1:U}$ is then fed to $\mathcal{J}$ together with $\mathbf{f}_{1:T}$ to generate an output distribution lattice $\mathbf{Z}$ of dimension $K \times T \times U$, where K is the output vocabulary size. The probability $p(k|t,u)$ of emitting token k at node (t,u) in $\mathbf{Z}$ is then defined as the k-th entry in the vector obtained after applying softmax to $\mathbf{Z}(t,u)$. Neural transducer training aims to maximise the probability $p(\mathbf{y}|\mathbf{X})$, which can be efficiently computed using a forward-backward algorithm on the lattice $\mathbf{Z}$ [19].

Neural transducers generally outperform CTC models due to their ability to model the dependency across output tokens. As the decoding procedure is frame-synchronous, transducers are suitable for streaming applications if the encoder has only a limited exposure to future acoustic context [20, 21].

3. KNOWLEDGE DISTILLATION FOR TRANSDUCERS

Knowledge distillation (KD) [9] is a widely used model compression technique that trains a small student model to match the output of a large teacher model. For a classification task with K classes, KD usually minimises the Kullback–Leibler (KL) divergence between the teacher output distribution $p_{\mathcal{T}}$ and the student output distribution $p_{\mathcal{S}}$ as given in Eqn. (2).

$$\mathcal{L}_{\text{KD}} = -\sum_{k=1}^{K} p_{\mathcal{T}}(k) \log \frac{p_{\mathcal{S}}(k)}{p_{\mathcal{T}}(k)}. \quad (2)$$

Since it does not require ground truth labels, KD can leverage unlabelled data for student model training. KD also provides greater flexibility for model compression since it does not constrain the input features or the model architectures of the teacher and student models to be the same.

3.1. KD for Non-Streaming Transducer

As described in Sec. 2.2, a neural transducer generates an output distribution lattice $\mathbf{Z}$ from an acoustic feature sequence $\mathbf{X}$ and a label sequence $\mathbf{y}$, from which the total conditional probability of the label sequence given the acoustic sequence can be computed by summing over all possible alignments. Therefore, the output lattice of a teacher transducer is an obvious distillation target for the student model. A straightforward distillation objective is to minimise the KL divergence (or cross-entropy as the teacher labels are fixed) between the output distribution lattices of the student $\mathbf{Z}_{\mathcal{S}}$ and teacher $\mathbf{Z}_{\mathcal{T}}$, which results in the loss function in Eqn. (3). As a result, the output distribution of the student model should imitate the teacher distribution over the entire lattice. However, this method is impractical due to its large memory and computation complexities of $\mathcal{O}(KTU)$.

$$\mathcal{L}_{\text{KD}} = -\sum_{t=1}^{T}\sum_{u=1}^{U}\sum_{k=1}^{K} \mathbf{Z}_{\mathcal{T}}(k,t,u) \log \mathbf{Z}_{\mathcal{S}}(k,t,u). \quad (3)$$

To reduce the computational cost of KD for transducers, distillation based on a collapsed distribution of all nodes in the lattice was proposed [22]. By only considering the blank symbol, the correct output symbol and all the remaining symbols, the output distribution for each node is reduced from K to only 3 classes. The complexity becomes $\mathcal{O}(TU)$. However, this method ignores the correlation across different output symbols compared to Eqn. (3), which could be important for training the student model using KD.

To achieve memory efficiency and also preserve the full distribution, we propose to distil knowledge only over the one-best path in the lattice instead of over the whole lattice. It has been shown that only a small proportion of nodes in the lattice contribute to the final decoding results [29], which indicates that a large number of paths in the lattice have low probabilities. As an approximation, only the nodes along the one-best path are considered as important. The distillation loss for one-best path in the output distribution lattice is

$$\mathcal{L}_{\text{KD}} \approx -\sum_{(t,u)\in 1\text{BEST}}\sum_{k=1}^{K} \mathbf{Z}_{\mathcal{T}}(k,t,u) \log \mathbf{Z}_{\mathcal{S}}(k,t,u), \quad (4)$$

where 1BEST is the most likely alignment between $\mathbf{X}$ and $\mathbf{y}$ containing the blank symbol. For an input utterance of length T and a target sequence of length U, the memory complexity is only $\mathcal{O}(K(T+U))$. Compared with [22], the proposed approach preserves the full distribution for each node at the expense of ignoring other possible alignments. Assuming the output lattice is relatively sparse, the proposed method is expected to transfer richer information than the collapsed distribution [22]. Furthermore, using the one-best alignment allows the alignment to be delayed when training a streaming student from a non-streaming teacher as discussed in Sec. 3.2.

3.2. KD for Streaming Transducer

In contrast to non-streaming transducers, directly applying one-best KD as in Eqn. (4) to streaming transducers may not be sensible. As little (or no) future context is available for a streaming transducer, it tends to emit symbols later than non-streaming models to gather more future information [30]. In other words, the most likely alignment for a streaming transducer normally lags behind its non-streaming counterpart by several time steps. As the pre-trained teacher model is non-streaming, directly applying Eqn. (4) to perform KD on a streaming student model may force the student model to "guess" into the future, which may lead to poor performance. To this end, a hyperparameter τ is introduced to delay the non-streaming alignment, which allows the student model to emit symbols τ frames later than its teacher. τ can be tuned to find the best trade-off between performance and latency. To train a streaming student model, the modified distillation loss is

$$\mathcal{L}_{\text{KD}} \approx - \sum_{(t,u)\in 1\text{Best}} \sum_{k=1}^{K} \mathbf{Z}_{\mathcal{T}}(k,t,u) \log \mathbf{Z}_{\mathcal{S}}(k,t+\tau,u). \quad (5)$$

3.3. Using Unlabelled Data and LM Fusion for KD

Unlike other model compression techniques, KD is a data-driven approach that is able to leverage unlabelled data to further improve the student model. For unlabelled speech data, the teacher model is used to generate one-best hypotheses via beam search. The one-best alignment and the output probabilities of each node on it for each utterance are used as distillation targets.

During decoding, an external LM can be used to improve the quality of the transcription [31, 32], which is beneficial to KD training. In this work, log-linear interpolation (shallow fusion) [31] of the transducer and an external LM is adopted to construct distillation targets containing LM information,

$$\mathbf{Z}'_{\mathcal{T}}(t,u) = \log(\text{Softmax}(\mathbf{Z}_{\mathcal{T}}(t,u)) + \beta \log(\text{LM}(t)), \quad (6)$$

where $\text{LM}(t)$ is a K-dimensional vector representing the LM output distribution given the previously predicted sequence before t and β is a tuned LM weight. Note that the LM score for the blank symbol is set to zero when the blank symbol is emitted and $\min(\text{LM}(t))$ when a non-blank symbol is emitted. For better convergence, the distillation loss is interpolated with the original transducer loss,

$$\mathcal{L} = \mathcal{L}_{\text{transducer}} + \lambda \mathcal{L}_{\text{KD}}, \quad (7)$$

where λ is the interpolation coefficient.

4. EXPERIMENTAL SETUP

4.1. Model

For the teacher model, the base w2v2 model [4] was used to initialise the encoder of a transducer model. The original w2v2 model takes the waveform as input and has an output frequency of 100 Hz, whereas filter bank features are more commonly used as ASR input and operate at 50 Hz. To achieve efficient knowledge distillation, a sub-sampling layer was added on top of the pre-trained model to reduce the frequency of the encoder output to 50 Hz. The student model is a small Conformer transducer model [24]. Both models have a single-layer long short-term memory (LSTM) projection network. The output vocabulary has 256 subword units generated using SentencePiece [33]. The details of both models are given in Table 1. Note that the student model has more than 10 times fewer parameters than the teacher model. For the LM used for shallow fusion, a 2-layer LSTM with 2048 units was trained from the LibriSpeech LM corpus. The vocabularies of the transducer models and the LM are the same. All models were implemented in ESPnet [34].

	Teacher model	Student model
Encoder	w2v2 [4]	Conformer-S [24]
Encoder dimension	768	144
Decoder dimension	640	320
Number of parameters	99.2M	9.7M

Table 1: Details of the teacher and student transducer models.

4.2. Data

The LibriSpeech dataset [35] was used for the experiments. The full training set has 960 hours of audiobook recordings. Among these, "train-clean-100" was used as labelled data while the remaining 860 hours were treated as unlabelled data. The teacher model was pre-trained using the full training set and fine-tuned on the 100-hour labelled subset following [4].

5. EXPERIMENTAL RESULTS

5.1. Baselines

Table 2 gives the word error rate (WER) of both the fine-tuned w2v2 models and the baseline conformer models trained on the LibriSpeech 100-hour subset. The fine-tuning configuration of the teacher model follows [4]. Although our implementation shares the same pre-trained encoder as the Base model in [4], the performance is better because our fine-tuned w2v2 model is a transducer model with BPE modelling units instead of a CTC model with grapheme units [4]. For the baseline Conformer-S models, SpecAugment [36] was used during training employing 2 frequency masks with $F = 27$ and 10 time masks with a maximum time-mask ratio of 0.05.

Models	dev		test	
	clean	other	clean	other
w2v2 CTC ([4])	6.1	13.5	6.1	13.3
w2v2 transducer (ours)	5.1	12.2	5.2	11.8
+ LM shallow fusion	4.2	10.4	4.3	10.1
Conformer-S (non-streaming)	7.5	20.9	8.0	21.8
Conformer-S (streaming)	11.2	28.4	12.2	29.6

Table 2: WERs of the w2v2 models and the baseline small Conformer models trained on LibriSpeech 100-hour subset. Our fine-tuned w2v2 transducer will be used as the teacher model.

As expected, the w2v2 transducer gives much lower WERs than the non-streaming Conformer-S model as it benefits from both self-supervised pre-training and a larger model size. For the streaming transducer, an attention mask was added to remove the contribution of future frames in the encoder and the 1D-convolution changed to 1D causal convolution to only attend to previous frames. Consequently, the WER for the streaming Conformer-S increases.

In the following experiments, the w2v2 transducer (2nd row in Table 2) was used as the teacher model. The student transducer model with the same architecture as the Conformer-S tries to approach the performance of the teacher model. For both the non-streaming and the streaming setups, the 100-hour labelled training data was used first to verify the proposed KD approach. Then, the remaining 860-hour training data was used as unlabelled data to further improve the student models.

5.2. KD for Non-Streaming Transducers

Two KD training strategies were investigated for the proposed distillation loss. The first strategy (ST1) trains the student transducer model from scratch using Eqn. (7). The second strategy (ST2) initialises the student model from a model trained on ground truth or pseudo transcriptions and then uses the proposed loss in Eqn. (7) for fine-tuning.

Transducer models	dev		test	
	clean	other	clean	other
Reference models: 100h labelled				
w2v2 transducer (teacher)	5.1	12.2	5.2	11.8
Conformer-S (baseline)	7.5	20.9	8.0	21.8
Student models: 100h labelled				
$\lambda = 0.001$, collapsed [22]	7.1	20.7	7.5	20.9
$\lambda = 0.001$, collapsed [22], ST2	6.8	19.2	7.2	19.8
$\lambda = 0.1$, ST1	7.1	19.8	7.6	20.5
$\lambda = 0.1$, ST2	6.7	19.0	7.1	19.3
$\lambda = 0.1$, ST2 [+LM]	6.7	18.9	7.0	19.3
Student models: 100h labelled + 860h unlabelled				
$\lambda = 0.0$	5.5	11.5	5.6	11.9
$\lambda = 0.0$ [+LM]	4.7	10.4	4.8	10.9
$\lambda = 0.1$, ST1	5.5	11.3	5.5	11.7
$\lambda = 0.1$, ST2	5.3	10.9	5.4	11.3
$\lambda = 0.1$, ST2 [+LM]	4.6	10.1	4.8	10.4

Table 3: WERs of non-streaming transducer models. λ is the weight of KD loss. [+LM] means either the teacher output distributions or the pseudo transcriptions are obtained with LM shallow fusion.

Five key observations can be made from Table 3. First, for student models trained from 100h of labelled data, the proposed distillation loss using the full distribution of the one-best alignment slightly outperforms KD using a collapsed distribution on the whole lattice [22]. Second, for student models trained from 100h labelled data and 860h unlabelled data, the proposed KD method yields lower WERs than simply using the pseudo transcriptions ($\lambda = 0.0$) of the unlabelled data because the output distribution carries more information from the teacher to the student than the pseudo transcriptions. Third, ST2 has lower WERs than ST1 for both the 100h and the 960h setups. As a result, ST2 will be used for the streaming experiments in Sec. 5.3. Fourth, the relative WER reductions (WERRs) on test-clean and test-other are 11.3% and 11.5% from using the proposed KD loss compared with the baseline. The WERRs increase to 32.5% and 48.2% when the additional 860h of unlabelled data was used for KD. Lastly, to distil knowledge from both the teacher model and the LM together, shallow fusion was used to generate the pseudo transcriptions on the unlabelled data and augment the output distributions. This leads to a WERR of 12.5% for the 100h setup compared with the baseline on test-clean. After incorporating the 860h of unlabelled data, the WERRs on test-clean and test-other improve to 40.0% and 52.3%.

5.3. KD for Streaming Transducers

By introducing the parameter τ for delaying the alignment, the proposed one-best distillation loss was applied to train streaming student models. As before, the distillation targets were still generated from the non-streaming teacher w2v2 transducer. When the streaming student model was trained to match the alignment of the non-streaming teacher, *i.e.* when $\tau = 0$, the WER of the student model is even higher than the baseline due to a lack of future context. After observing the one-best alignment of a streaming transducer can be a number of steps behind the one-best alignment of a non-streaming transducer, the WERs for $\tau = 6$ and $\tau = 7$ (equivalent to a look-ahead of 240 ms and 280 ms) are shown in Table 4. Although more delay steps result in improved WERs, the latency of the ASR model also increases. Therefore, setting a sensible value of τ is key to training a student model using KD. Compared to the streaming baseline, the proposed distillation approach achieves WERRs of 7.4% and 6.8% on test-clean and test-other in 100h experiments. With the additional 860h unlabelled data, the student model trained with KD outperforms the one trained only with pseudo transcriptions and increases WERRs to 19.7% and 38.5% w.r.t. the streaming baseline.

Transducer models	dev		test	
	clean	other	clean	other
Reference models: 100h labelled				
w2v2 transducer (teacher)	5.1	12.2	5.2	11.8
Conformer-S (baseline)	11.2	28.4	12.2	29.6
Student models: 100h labelled				
$\lambda = 0.1$, τ=0, ST2	20.2	39.7	20.5	41.4
$\lambda = 0.1$, τ=6, ST2	10.7	26.7	11.3	28.0
$\lambda = 0.1$, τ=7, ST2	10.6	26.4	11.3	27.6
Student models: 100h labelled + 860h unlabelled				
$\lambda = 0.0$	9.5	19.3	10.6	19.7
$\lambda = 0.1$, τ=7, ST2	9.2	18.3	9.8	18.2

Table 4: WERs of streaming transducer models. λ is the weight of the KD loss. τ is the number of delayed steps for the one-best alignment from the teacher model.

6. CONCLUSIONS

In this paper, we propose a simple and efficient knowledge distillation (KD) loss for neural transducers using the full distribution along the one-best alignment of the output distribution lattice. This approach can be easily extended to distil knowledge from a non-streaming transducer to a streaming one. We showed the effectiveness of the proposed approach by distilling knowledge from a non-streaming teacher model, initialised from a large self-supervised pre-trained model, to both non-steaming and streaming student models that are more than 10 times smaller. Experiments also show that the performance of the student model can be greatly improved by having more unlabelled data for KD or augmenting the distillation targets by language model shallow fusion. With better teacher models or more unlabelled data, the student models would be expected to reach better performance. In future, we will explore how to best transfer information from an external language model to the student model, and how to distil knowledge more effectively from a non-streaming teacher to a streaming student.

7. REFERENCES

[1] J. Devlin, M.W. Chang, K. Lee, & K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," *Proc. NAACL*, Minneapolis, 2019.

[2] T. Brown, B. Mann, N. Ryder, M. Subbiah, J.D. Kaplan, & Dhariwal, "Language models are few-shot learners," *Proc. NeurIPS*, Vancouver, 2020.

[3] A. Baevski, S. Schneider, & M. Auli, "vq-wav2vec: Self-supervised learning of discrete speech representations," *Proc. ICLR*, New Orleans, 2019.

[4] A. Baevski, Y. Zhou, A. Mohamed, & M. Auli, "wav2vec 2.0: A framework for self-supervised learning of speech representations," *Proc. NeurIPS*, Vancouver, 2020.

[5] Z. Peng, A. Budhkar, I. Tuil, J. Levy, P. Sobhani, R. Cohen, & J. Nassour, "Shrinking bigfoot: Reducing wav2vec 2.0 footprint," *arXiv preprint arXiv:2103.15760*, 2021.

[6] J. Xue, J. Li, & Y. Gong, "Restructuring of deep neural network acoustic models with singular value decomposition," *Proc. Interspeech*, Lyon, 2013.

[7] R. Prabhavalkar, O. Alsharif, A. Bruguier, & I. McGraw, "On the compression of recurrent neural networks with an application to LVCSR acoustic modeling for embedded speech recognition," *Proc. ICASSP*, Shanghai, 2016.

[8] K. Zhen, H.D. Nguyen, F.J. Chang, A. Mouchtaris, & A. Rastrow, "Sparsification via compressed sensing for automatic speech recognition," *Proc. ICASSP*, Toronto, 2021.

[9] G. Hinton, O. Vinyals, & J. Dean, "Distilling the knowledge in a neural network," *Proc. NIPS Deep Learning Workshop*, Montreal, 2014.

[10] S. Cao, Y. Kang, Y. Fu, X. Xu, S. Sun, *et al.*, "Improving streaming transformer based ASR under a framework of self-supervised learning," *Proc. Interspeech*, Brno, 2021.

[11] T. Doutre, W. Han, M. Ma, Z. Lu, C.C. Chiu, *et al.*, "Improving streaming automatic speech recognition with non-streaming model distillation on unsupervised data," *Proc. ICASSP*, Toronto, 2021.

[12] J. Li, R. Zhao, J.T. Huang, & Y. Gong, "Learning small-size DNN with output-distribution-based criteria," *Proc. Interspeech*, Singapore, 2014.

[13] J.H.M. Wong & M.J.F. Gales, "Sequence student-teacher training of deep neural networks," *Proc. Interspeech*, San Francisco, 2016.

[14] R. Takashima, S. Li, & H. Kawai, "An investigation of a knowledge distillation method for CTC acoustic models," *Proc. ICASSP*, Alberta, 2018.

[15] G. Kurata & K. Audhkhasi, "Improved knowledge distillation from bi-directional to uni-directional LSTM CTC for end-to-end speech recognition," *Proc. SLT*, Athens, 2018.

[16] R. Takashima, L. Sheng, & H. Kawai, "Investigation of sequence-level knowledge distillation methods for CTC acoustic models," *Proc. ICASSP*, Brighton, 2019.

[17] J. Yoon, H. Lee, H.Y. Kim, W.I. Cho, & N.S. Kim, "TutorNet: Towards flexible knowledge distillation for end-to-end speech recognition," *IEEE/ACM Trans. on Audio, Speech, and Language Processing*, vol. 29, 2021.

[18] R. Pang, T. Sainath, R. Prabhavalkar, S. Gupta, & C.C. Chiu, "Compression of end-to-end models," *Proc. Interspeech*, Hyderabad, 2018.

[19] A. Graves, "Sequence transduction with recurrent neural networks," *Proc. ICML Workshop on Representation Learning*, Edinburgh, 2012.

[20] K. Rao, H. Sak, & R. Prabhavalkar, "Exploring architectures, data and units for streaming end-to-end speech recognition with RNN-transducer," *Proc. ASRU*, Okinawa, 2017.

[21] Y. He, T.N. Sainath, R. Prabhavalkar, I. McGraw, R. Alvarez, *et al.*, "Streaming end-to-end speech recognition for mobile devices," *Proc. ICASSP*, Brighton, 2019.

[22] S. Panchapagesan, D.S. Park, C.C. Chiu, Y. Shangguan, Q. Liang, & A. Gruenstein, "Efficient knowledge distillation for RNN-transducer models," *Proc. ICASSP*, Toronto, 2021.

[23] R.V. Swaminathan, B. King, G.P. Strimel, J. Droppo, & A. Mouchtaris, "CoDERT: Distilling encoder representations with co-learning for transducer-based speech recognition," *Proc. Interspeech*, Brno, 2021.

[24] A. Gulati, J. Qin, C.C. Chiu, N. Parmar, Y. Zhang, *et al.*, "Conformer: Convolution-augmented transformer for speech recognition," *Proc. Interspeech*, Shanghai, 2020.

[25] D. Jiang, X. Lei, W. Li, N. Luo, Y. Hu, *et al.*, "Improving transformer-based speech recognition using unsupervised pre-training," *Proc. Interspeech*, Graz, 2019.

[26] Y. Zhang, J. Qin, D.S. Park, W. Han, C. Chiu, *et al.*, "Pushing the limits of semi-supervised learning for automatic speech recognition," *Proc. NeurIPS SAS Workshop*, Vancouver, 2020.

[27] A. Graves, S. Fernández, F. Gomez, & J. Schmidhuber, "Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks," *Proc. ICML*, Pittsburgh, 2006.

[28] Q. Zhang, H. Lu, H. Sak, A. Tripathi, E. McDermott, *et al.*, "Transformer transducer: A streamable speech recognition model with transformer encoders and RNN-T loss," *Proc. ICASSP*, 2020.

[29] M. Jain, K. Schubert, J. Mahadeokar, C.F. Yeh, K. Kalgaonkar, *et al.*, "RNN-T for latency controlled ASR with improved beam search," *arXiv.org*, 1911.01629, 2019.

[30] G. Kurata & G. Saon, "Knowledge distillation from offline to streaming RNN transducer for end-to-end speech recognition," *Proc. Interspeech*, Shanghai, 2020.

[31] D. Bahdanau, J. Chorowski, D. Serdyuk, P. Brakel, & Y. Bengio, "End-to-end attention-based large vocabulary speech recognition," *Proc. ICASSP*, Shanghai, 2016.

[32] A. Kannan, Y. Wu, P. Nguyen, T.N. Sainath, Z. Chen, & R. Prabhavalkar, "An analysis of incorporating an external language model into a sequence-to-sequence model," *Proc. ICASSP*, Alberta, 2018.

[33] T. Kudo & J. Richardson, "SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing," *Proc. EMNLP*, Brussels, 2018.

[34] S. Watanabe, T. Hori, S. Karita, T. Hayashi, J. Nishitoba, *et al.*, "ESPnet: End-to-end speech processing toolkit," *Proc. Interspeech*, Hyderabad, 2018.

[35] V. Panayotov, G. Chen, D. Povey, & S. Khudanpur, "LibriSpeech: An ASR corpus based on public domain audio books," *Proc. ICASSP*, Brisbane, 2015.

[36] D.S. Park, W. Chan, Y. Zhang, C.C. Chiu, B. Zoph, *et al.*, "SpecAugment: A simple data augmentation method for automatic speech recognition," *Proc. Interspeech*, Graz, 2019.