KNOWLEDGE TRANSFER FROM LARGE-SCALE PRETRAINED LANGUAGE MODELS TO END-TO-END SPEECH RECOGNIZERS

Yotaro Kubo, Shigeki Karita, Michiel Bacchiani

Google
Shibuya Stream, 3–21–3 Shibuya, Shibuya-ku, Tokyo.

ABSTRACT

End-to-end speech recognition is a promising technology for enabling compact automatic speech recognition (ASR) systems since it can unify the acoustic and language model into a single neural network. However, as a drawback, training of end-to-end speech recognizers always requires transcribed utterances. Since end-to-end models are also known to be severely data hungry, this constraint is crucial especially because obtaining transcribed utterances is costly and can possibly be impractical or impossible. This paper proposes a method for alleviating this issue by transferring knowledge from a language model neural network that can be pretrained with text-only data. Specifically, this paper attempts to transfer semantic knowledge acquired in embedding vectors of large-scale language models. Since embedding vectors can be assumed as implicit representations of linguistic information such as part-of-speech, intent, and so on, those are also expected to be useful modeling cues for ASR decoders. This paper extends two types of ASR decoders, attention-based decoders and neural transducers, by modifying training loss functions to include embedding prediction terms. The proposed systems were shown to be effective for error rate reduction without incurring extra computational costs in the decoding phase.

Index Terms— Automatic speech recognition, knowledge transfer, multi-task learning, pretrained language models, BERT

1. INTRODUCTION

End-to-end techniques are known to be beneficial for realizing compact models for automatic speech recognition (ASR). Since the compact models are advantageous both in terms of interface responsiveness and energy consumption, an accuracy improvement in end-to-end ASR is a crucial step for expanding the availability of speech recognition technologies in, for example mobile devices [1].

One of the most promising approaches for improving end-to-end ASR accuracy is introducing additional training data since end-to-end models typically require large amount of training data for mitigating over-fitting. However, obtaining labeled training data is often costly compared to obtaining unpaired audio-only and/or text-only data. End-to-end systems model speech recognition as one unified model as opposed to a conventional system that factors the problem into several models and separates the language model (LM) component from the acoustic model (AM) component. Hence, training from text-only data can only be realized in the form of an external LM which leads to additional complexity and run-time costs.

Recently, a series of important methods for utilizing unpaired audio-only data is proposed as pretraining methods of the end-to-end models [2, 3, 4]. Those methods pretrain the encoder part of end-to-end ASR models by introducing a surrogate objective function based on mutual information between instantaneous feature representation and global context representation. Since those pretraining processes can be performed with audio-only data, those methods are promising for relaxing the limitation due to availability of paired training data.

Previous work has shown an effective way to make use of unpaired audio data, the work here focus on a way to make use of unpaired text data without raising the model size and inference complexity. There are several methods for integrating external language models trained on text-only data into end-to-end speech recognition [5]. However, since language models are usually very large in terms of model size and computational complexity, the use of external LMs makes the model less compact and raises the computation cost. Recent language models, such as *bidirectional encoder representations from transformers* (BERT) [6], involve typically several hundred millions of parameters, and integrating it is computationally prohibitive for small devices even though the language model performs well and is in that respect desirable.

In this paper, learning from text-only data is achieved by designing multi-task learning that performs knowledge distillation [7]. Applications of transfer learning do not require any additional computational cost in the recognition phase, and thus it is advantageous for models being deployed to small devices. There are several recent attempts to distill knowledge from external models into end-to-end speech recognizers [8, 9, 10, 11]. The prior studies that are most related to our approach are based on distillation from a masked language model [10, 11]. For transferring knowledge from a pretrained BERT model, [10] minimizes the KL-divergence between the token posterior distribution from the BERT masked language model and the ASR model. In contrast, our approach focuses on using embedding vectors rather than posterior distribution of the masked LM. As suggested in the experiments in the original BERT paper [6], embedding vectors are believed to be a compact representation containing richer information compared to the posterior distributions of tokens. On the other hand, knowledge transfer from embedding vectors is shown to be effective for improving performance of non-auto-regressive speech recognizers [11]. In this paper, the effectiveness of knowledge transfer from embedding vectors is verified with attention- [12] and transducer-based [13] speech recognition models. Since the transducer-based approach is popular in ASR for computationally limited devices as described in [1], it is important to analyze the effectiveness of knowledge transfer for such models.

2. KNOWLEDGE DISTILLATION FROM EMBEDDINGS

This section describes methods to enhance existing end-to-end ASR neural networks to multitask models that predict precomputed token-embedding vectors in addition to the token distributions. In this section, each element in the training dataset is denoted as a triple $(\mathbf{X}, \boldsymbol{y}, \mathbf{E})$ of the feature vector sequence $\mathbf{X}$, token sequence $\boldsymbol{y} =$

$(y_1, y_2, \cdots)$, and the sequence of token embedding vectors $\mathbf{E} = (\boldsymbol{e}_1, \boldsymbol{e}_2, \cdots)$. Each token $y_i \in V$ represents a subword token, typically a word-piece [14], of the transcription. The embedding vectors $\mathbf{E}$ and the token sequences $\boldsymbol{y}$ are related by a word embedding function $\mathcal{M} : V^N \to (\mathbb{R}^{D^{\text{Emb}}})^N$ as $\mathbf{E} = \mathcal{M}(\boldsymbol{y})$, and therefore the numbers of word tokens and the token-embedding vectors in the same triple are the same. In this paper, $\mathcal{M}$ is represented in a pretrained neural network, and its parameters are not fine-tuned. Specifically, in the experimental section below, we used BERT embeddings as $\mathcal{M}$.

2.1. Attention-based decoder with auxiliary regression

An attention-based auto-regressive decoder predicts the probability distribution of the next token y_i using the current decoder state $\boldsymbol{\phi}(\mathbf{X}, \boldsymbol{y}_{1:i-1})$ as,

$$p(y_i \mid \mathbf{X}, \boldsymbol{y}_{1:i-1}) \propto \exp\left(\boldsymbol{\lambda}_{y_i}^\mathsf{T} \boldsymbol{\phi}(\mathbf{X}, \boldsymbol{y}_{1:i-1})\right). \quad (1)$$

Here, $\boldsymbol{\lambda}_{y_i}$ is the parameter vector used for computing a logit for y_i, and $\boldsymbol{\phi}(\mathbf{X}, \boldsymbol{y}_{1:i-1}) \in \mathbb{R}^D$ is a vector referred to as the "pre-softmax activation". Typically, $\boldsymbol{\phi}$ is equipped with an attention mechanism to summarize the encoded representation of $\mathbf{X}$.

A cross-entropy loss is defined by using the above prediction as,

$$L^{\text{XEnt}}(\mathbf{X}, \boldsymbol{y}) = -\sum_i \log p(y_i \mid \mathbf{X}, \boldsymbol{y}_{1:i-1}). \quad (2)$$

The conventional training for end-to-end models with an attention-based decoder minimizes the expectation of this objective function using a gradient-based method such as Adam [15].

In this paper, aiming at transferring knowledge from token-embedding vectors to the decoder state vector $\boldsymbol{\phi}$, a regression neural network $\mathcal{R} : \mathbb{R}^D \to \mathbb{R}^{D^{\text{Emb}}}$ is introduced. With this additional component, the following loss function L^{Emb} based on the distances between the estimated embeddings and the precomputed embeddings $\boldsymbol{e}_i$ is introduced as an auxiliary loss function to be minimized.

$$L^{\text{Emb}}(\mathbf{X}, \boldsymbol{y}, \mathbf{E}) = \sum_i d(\mathcal{R}(\boldsymbol{\phi}(\mathbf{X}, \boldsymbol{y}_{1:i-1})), \boldsymbol{e}_i), \quad (3)$$

where d is a differentiable distance function such as Lp distances.

With the above auxiliary loss function, the loss function to be minimized in training can be defined as,

$$L^{\text{XEnt}}(\mathbf{X}, \boldsymbol{y}) + \sigma L^{\text{Emb}}(\mathbf{X}, \boldsymbol{y}, \mathbf{E}). \quad (4)$$

Here, σ is a hyper-parameter that controls the relative importance of the auxiliary task. Setting $\sigma = 0$ reduces the training process to the conventional cross-entropy on the token probabilities alone.

Fig 1-(a) depicts neural net architecture for implementing training with this loss function. The "Regression Net" and "Pretrained Embedding Model" components in the figure can be pruned in the recognition phase. Therefore, the total computational complexity at the recognition phase is not increased by this modification.

2.2. Transducer-based decoder with auxiliary regression

Neural transducers (also known as recurrent neural network transducers (RNN-Ts)) [13] model a probability distribution of alignment variables $\boldsymbol{z} = (z_1, z_2, \cdots) \in (V \cup \{\varphi\})^{T+N}$ where T and N are the lengths of a feature vector and a word sequence, respectively, and φ is a blank symbol. The token sequence $\boldsymbol{y}$ is considered to be extracted by using a *blank removal function* $\mathcal{B}$ that removes φ from the alignment sequence $\boldsymbol{z}$. Therefore, the predictive distribution of the token sequences can be expressed by marginalizing the alignment variables, as follows:

$$p(\boldsymbol{y} \mid \mathbf{X}) = \sum_{\boldsymbol{z}'} \mathbf{1}\left[\mathcal{B}(\boldsymbol{z}') = \boldsymbol{y}\right] p(\boldsymbol{z}' \mid \mathbf{X}). \quad (5)$$

Here, $\mathbf{1}[P]$ is an indicator function whose value is 1 if the predicate P is true, and 0 otherwise. It should be noted that $\mathbf{1}[\mathcal{B}(\boldsymbol{z}') = \boldsymbol{y}]$ can be viewed as a hard-coded model of $p(\boldsymbol{y} \mid \boldsymbol{z}')$.

The probability distribution over the alignment $p(\boldsymbol{z} \mid \mathbf{X})$ is computed depending on outputs of two neural networks, the transcription and prediction networks. The transcription network computes feature vectors $\boldsymbol{\phi}_t$ for each frame t. The prediction network computes the representation from the prefix of a (hypothesis) token sequence. In the training phase with teacher forcing, the prediction network computes output vectors $\boldsymbol{\psi}_i$ for each token y_i using the token prefix $\boldsymbol{y}_{1:i-1}$. In this paper, $\boldsymbol{\phi}_t$ and $\boldsymbol{\psi}_i$ are called acoustic and language feature vectors.

Using those two feature vectors, the joint network $\mathcal{J}$ computes the probability distribution over the alignment variable $\boldsymbol{z}$ as,

$$p(z_n = k \mid \boldsymbol{z}_{1:n-1}, \mathbf{X}, \boldsymbol{y}) = \mathcal{J}_k(\boldsymbol{\phi}_{1+\tau(\boldsymbol{z}_{1:n-1})}, \boldsymbol{\psi}_{1+\iota(\boldsymbol{z}_{1:n-1})}), \quad (6)$$

where $\mathcal{J}_k(.)$ is the k-th element of the output vector of the joint network, $\tau(\boldsymbol{z}_{1:n-1})$ and $\iota(\boldsymbol{z}_{1:n-1})$ are the numbers of blank φ and non-blank symbols, respectively, in the prefix $\boldsymbol{z}_{1:n-1}$.

Joint regression network

Here, similar to attention-based decoder modeling, a regression network $\mathcal{R}$ is introduced to predict token-embedding vectors $\mathbf{E}$. The auxiliary loss can be defined as an expectation of distances as,

$$L^{\text{Emb}}(\mathbf{X}, \boldsymbol{y}, \mathbf{E}) = \sum_i \langle d(\mathcal{R}(\boldsymbol{\phi}_t, \boldsymbol{\psi}_i), \boldsymbol{e}_i) \rangle_{q_i(t \mid \mathbf{X}, \boldsymbol{y})}. \quad (7)$$

It should be noted that the regression network here takes two input vectors similar to the joint network. The alignment probability $q_i(t \mid \mathbf{X}, \boldsymbol{y})$ is a probability of t-th acoustic feature vector being consumed after processing the prefix $\boldsymbol{y}_{1:i-1}$, and can be expressed as,

$$q_i(t \mid \mathbf{X}, \boldsymbol{y}) \overset{\text{def}}{=} \sum_{\boldsymbol{z}'} \sum_n \mathbf{1}\left[\iota(\boldsymbol{z}'_{1:n}) = i - 1 \wedge \tau(\boldsymbol{z}'_{1:n}) = t - 1\right] p(\boldsymbol{z}' \mid \mathbf{X}, \boldsymbol{y}). \quad (8)$$

The expectation over this posterior distribution can be computed by reusing the results of the forward-backward algorithm used in the computation of the conventional transducer loss function. It should be noted that gradient information was not propagated through this q function in the experimental section below, to ensure that auxiliary loss does not affect to alignment computation.

Fig 1-(b) depicts an example neural net architecture for this training. The expectation in Eq. (7) can efficiently be computed by reusing the alignment variable (specifically, forward and backward score) from the forward-backward computation in the main transducer loss.

Token-synchronous regression network

As an alternative to the loss function described above, we investigated another loss that has a single regression result per token.
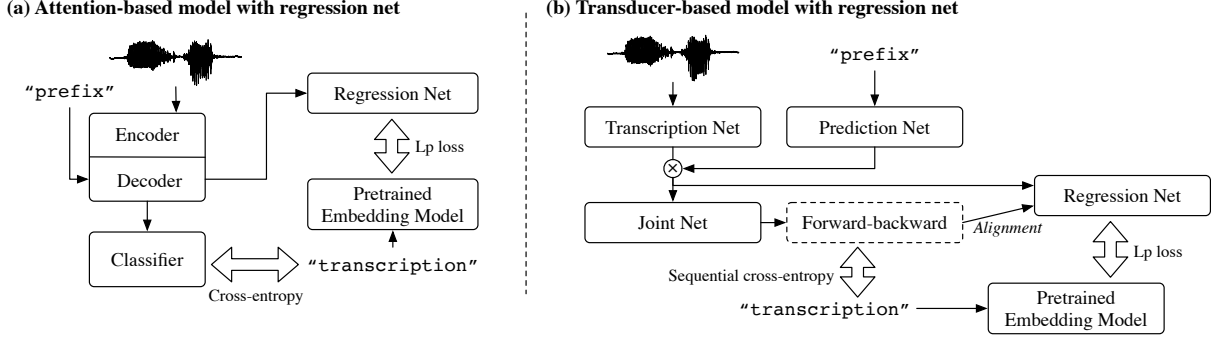
(a) Attention-based model with regression net

(b) Transducer-based model with regression net

Fig. 1. Block diagrams of (a) attention-based model with a regression net, and (b) transducer-based model with a regression net.

This variant approximates the expectation of the loss functions in Eq. (7) by using the expectation of the acoustic features as,

$$L^{\text{Emb}}(\mathbf{X}, \boldsymbol{y}, \mathbf{E}) \simeq \sum_i d(\mathcal{R}(\langle \boldsymbol{\phi}_t \rangle_{q_i(t|\mathbf{X},\boldsymbol{y})}, \boldsymbol{\psi}_i), \boldsymbol{e}_i). \qquad (9)$$

This approximated loss is equivalent to Eq. (7) when an L2 distance metric and a linear network are used as d and $\mathcal{R}$, respectively.

This approximation reduces the training memory consumption regarding the regression network from $O(TN)$ to $O(N)$. Furthermore, the regression network in this architecture is trained to make a single prediction per token whereas the joint regression network in Eq. (7) makes multiple predictions minimizing the expectation of the loss. Therefore, this approximation is also beneficial as the token-wise predictions can be helpful for subsequent processing of the ASR results.

3. EXPERIMENTS

We verified the effectiveness of the proposed methods by training models on the LibriSpeech dataset [16] (960h of transcribed speech), and using a BERT [6] language model pretrained with the BooksCorpus (800M words) [17] and Wikipedia text data (2500M words).

3.1. Experimental Setup

The attention-based model we used was based on bi-directional long short-term memory (BLSTMs). The configurations for the BLSTM encoder and decoder were set to be identical with those of [18]. The transducer-based model employed a Conformer encoder [19]. The encoder consisted of 17 Conformer blocks, and each block was equipped with an 8-head dot-attention, 512-channel 1d-convolution, and 2048-dim feed-forward module. The prediction network in the transducer-based model was an LSTM consisting of a 640-dimensional (uni-directional) long short-term memory (LSTM) layer and a 128-dimensional token-embedding layer. The joint network in the transducer-based model was a feed-forward network with a 640-dim hidden layer with $\tanh$-activation.

SpecAugment [20] was applied in the transducer-based systems. The numbers of time- and frequency-masks were set to 10 and 2, respectively. The lengths of the time- and frequency-masks were set to $0.05T$ and 27, respectively, where T is the length of the input feature vector sequence. Both for the attention and transducer-based models, Adam [15] was used as the optimization method. The batch sizes were set to 512 and 2048 for the attention and transducer-based models, respectively. The optimization results of transducer-based systems was smoothed by computing exponential moving average

(EMA) of the parameter trajectory. The decay rate for EMA was set to 0.9999. For each training configuration, the best checkpoint was chosen based on the WERs observed on the devother partition.

The weight for the auxiliary task (σ in Eq. (4)) was varied for analyzing the effect of the auxiliary task. For attention-based models, the weights were selected from $\sigma \in \{2^0, 2^{-1}, 2^{-2}, 2^{-3}, 2^{-4}\}$, and for transducer-based models, the weights were selected from $\sigma \in \{10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}\}$. As it is mentioned above, setting $\sigma = 0$ is equivalent with omitting the auxiliary task, and was considered as a baseline setting.

The distance metric used in the auxiliary loss function was chosen as the L1 distance between the predicted and target vectors normalized (divided) by the dimensionality of the embedding vectors ($D^{\text{Emb}} = 768$). The regression network $\mathcal{R}$ was defined by a single affine transformation. For the transducer-based decoder, we first used a token-synchronous regression module, i.e. the loss function in Eq. (9). As described in the previous section, Eqs. (9) and (7) are identical if we use an L2 distance and a linear regression network. We used an L1 distance in this experiment and expected a subtle difference which we analyzed further in the results below.

The pretrained BERT model we used is identical with one available on TFHub [21]. This BERT embedding module consists of 12 transformer blocks with 768 hidden units and 12 self-attention heads. For generating the target embedding vector sequences for training, the transcriptions in the training data were tokenized and fed into the BERT embedding module. In this experiment, since we adopted BERT tokens using 30k word-piece vocabulary, the ASR models were also trained with this tokenization method. Since word-piece models used in the baseline ASR model did not match those used for BERT modeling, we also investigated the effect of using word-piece models developed for BERT in ASR [1].

3.2. Main results

Table 1 shows the word error rates (WERs) of the baseline and the proposed training methods applied to attention-based models. By comparing "Baseline" and "BERT tok.", we observe that the use of a different tokenization strategy did not degrade the word error rates significantly. By increasing, the auxiliary task weight σ, we observed a word error rate reduction in all the test sets. The best configuration for all the test sets is obtained when σ was set to 2^{-2} where the relative error rate reduction was -16.7% and -10.6% for the testclean and testother datasets. By increasing the auxiliary weight further, we observed a performance degradation.

---

[1] We kept special tokens for BERT modeling, such as [CLS] and [SEP] as ASR output target sequence.

LAS		dev		test	
	σ	clean	other	clean	other
Baseline	–	4.5	13.5	4.7	13.7
BERT tok.	0	4.6	13.2	4.8	14.1
+BERT reg.	0.0625	4.2	12.1	4.3	13.2
	0.125	4.0	11.8	4.2	12.7
	0.25	**3.9**	**11.6**	**4.0**	**12.6**
	0.5	4.1	12.3	4.2	13.2
	1.0	4.2	13.0	4.2	13.8

Table 1. WERs of attention-based models: "BERT tok." denotes a system with the baseline training method and word-piece model from pretrained BERT model. "+BERT reg." denotes the systems with auxiliary regression task with varying task weights.

ConformerS		dev		test	
	σ	clean	other	clean	other
Baseline	–	2.6	6.8	2.7	6.3
BERT tok.	0	2.6	5.7	2.7	6.2
+BERT reg.	0.0001	2.5	5.5	2.6	5.6
	0.001	2.4	**5.2**	2.7	5.7
	0.01	**2.4**	5.3	**2.6**	**5.5**
	0.1	2.4	5.3	2.8	5.9

Table 2. WERs of transducers (see also Table 1 for legends.)

Table 2 shows the word error rates of the baseline and the proposed training methods applied to transducer-based models. The best configuration for `devclean` is obtained when σ was set to 10^{-2} where the relative error rate reduction was -3.7% and -11.3% for `testclean` and `testother` datasets.

From both applications, we confirmed that the multitask learning with an auxiliary regression to the word embeddings were beneficial for transferring semantic knowledge from the pretrained BERT language model. Since setting the auxiliary weight σ to zero converges to the single task model, we observed WER convergence to the baseline model by reducing σ. We also observed subtle differences due to the tokenization method we employed. The WordPiece models we took from the BERT model performed better with our Conformer models. This might be due to the fact that the Conformer has more expressive ability and the larger inventory of tokens in BERT-based tokenization worked well with this setting.

3.3. Comparison between token-synchronous regression and joint regression net

In the previous section, we verified the effectiveness of our method applied to transducer-based decoders using token-synchronized regression shown in Eq. (9). Even though the difference between the two variants proposed in Section 2.2 is subtle, they are not identical in this experiment since an L1 distance is used as the distance metric.

Table 3 summarizes the results of the model with joint regression and with token-synchronized regression networks. The best auxiliary weight was chosen based on the results on `devother`. From the table, we could not observe clear performance difference between the loss functions compared (Eqs. (7) and (9)).

As mentioned in Section 2.2, the token-synchronized regression network has qualitative advantages. Therefore, in this setting, we confirmed that the use of token-synchronized regression network was more relevant. However, the accuracy difference might be larger if we were to use more complicated regression network than a linear

ConformerS		dev		test	
	Best σ	clean	other	clean	other
BERT tok.	0	2.6	5.7	2.7	6.2
Token-sync. regr.	0.01	2.4	5.3	2.6	5.5
Joint regr.	0.01	2.5	5.1	2.7	5.8

Table 3. Comparison between token-synchronized regression and joint regression networks.

Pretrained-ConfXL		dev		test	
	Best σ	clean	other	clean	other
Baseline	0	1.77	3.47	1.75	3.56
+BERT reg.	0.01	1.73	3.40	1.80	3.51

Table 4. WERs with pretrained acoustic encoders

regression network.

3.4. Results with pretrained acoustic encoders

As an additional experiment, we combined the proposed training method with a large-scale pretrained acoustic encoder. Application for a model like that is most promising as it opens the door for end-to-end models to use both unpaired audio and text data. Specifically, we evaluated the proposed methods with Conformer-based wav2vec-2.0 pretraining method [4]. The pretrained encoder and the decoder setting were identical with those of "Conformer-XL" in [4] pretrained with LibriLight dataset (60k hours). As in the previous experiments, the hyper-parameters were chosen to minimize the word-error rates over the `devother` test set.

Table 4 shows the WERs of the proposed training methods combined with the pretrained encoder. We confirmed that the proposed method could further reduce the word error rate even from the strong baseline with a pretrained encoders. However, the advantage is relatively small and this suggest that a part of semantic information might already be captured in wav2vec-2.0 training even without using a transcription. Future work will focus on improving the configuration for this type of combination.

4. CONCLUSION

This paper proposed a simple method to utilize text-only data and pretrained word embedding models for enhancing performance of speech recognition without increasing the model size. The proposed method is based on multi-task learning consisting of a main speech recognition task and an auxiliary word-embedding regression task. The multi-task learning method is applied to both attention and transducer-based end-to-end speech recognizers. We confirmed that, with including the auxiliary task as our methods propose, we were able to improve the transcription accuracies of the resulting systems. For attention-based models, the error reductions were 16.7% and 10.6% for the `testclean` and `testother` datasets, respectively. For transducer-based models, the error reductions were 3.7% and 11.3% for the `testclean` and `testother` datasets, respectively.

Future work will focus on the usability of token-embeddings, obtained as a by-product of this model, for down-stream natural language processing tasks. Even though we confirmed that a training process with the proposed method can reduce the distance metric between the estimated and precomputed token-embedding vectors, it is not clear if the estimated vectors are beneficial for, for example, an utterance classification task.

5. REFERENCES

[1] Yanzhang He, Tara N Sainath, Rohit Prabhavalkar, Ian Mc-Graw, Raziel Alvarez, Ding Zhao, David Rybach, Anjuli Kannan, Yonghui Wu, Ruoming Pang, et al., "Streaming end-to-end speech recognition for mobile devices," in *Proc. ICASSP*, 2019, pp. 6381–6385.

[2] Aäron van den Oord, Yazhe Li, and Oriol Vinyals, "Representation learning with contrastive predictive coding," *arXiv preprint arXiv:1807.03748*, 2018.

[3] Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli, "wav2vec 2.0: A framework for self-supervised learning of speech representations," in *Proc. NeurIPS*, 2020.

[4] Yu Zhang, James Qin, Daniel S Park, Wei Han, Chung-Cheng Chiu, Ruoming Pang, Quoc V Le, and Yonghui Wu, "Pushing the limits of semi-supervised learning for automatic speech recognition," *arXiv preprint arXiv:2010.10504*, 2020.

[5] Shubham Toshniwal, Anjuli Kannan, Chung-Cheng Chiu, Yonghui Wu, Tara N Sainath, and Karen Livescu, "A comparison of techniques for language model integration in encoder-decoder speech recognition," in *Proc. IEEE Workshop on SLT*, 2018, pp. 369–375.

[6] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proc. NAACL-HLT, Vol. 1*, 2019, pp. 4171–4186.

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

[8] Gakuto Kurata and George Saon, "Knowledge distillation from offline to streaming RNN transducer for end-to-end speech recognition," in *Proc. INTERSPEECH*, 2020, pp. 2117–2121.

[9] Sankaran Panchapagesan, Daniel S Park, Chung-Cheng Chiu, Yuan Shangguan, Qiao Liang, and Alexander Gruenstein, "Efficient knowledge distillation for RNN-transducer models," in *Proc. ICASSP*, 2021, pp. 5639–5643.

[10] Hayato Futami, Hirofumi Inaguma, Sei Ueno, Masato Mimura, Shinsuke Sakai, and Tatsuya Kawahara, "Distilling the knowledge of BERT for sequence-to-sequence ASR," in *Proc. INTERSPEECH*. 2020, pp. 3635–3639, ISCA.

[11] Ye Bai, Jiangyan Yi, Jianhua Tao, Zhengkun Tian, Zhengqi Wen, and Shuai Zhang, "Fast end-to-end speech recognition via non-autoregressive models and cross-modal knowledge transferring from BERT," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 29, pp. 1897–1911, 2021.

[12] William Chan, Navdeep Jaitly, Quoc Le, and Oriol Vinyals, "Listen, attend and spell: A neural network for large vocabulary conversational speech recognition," in *Proc. ICASSP*, 2016, pp. 4960–4964.

[13] Alex Graves, "Sequence transduction with recurrent neural networks," in *Proc. ICML Representation Learning Workshop*, 2012.

[14] Mike Schuster and Kaisuke Nakajima, "Japanese and Korean voice search," in *Proc. ICASSP*, 2012, pp. 5149–5152.

[15] Diederik P Kingma and Jimmy Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[16] Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur, "Librispeech: An asr corpus based on public domain audio books," in *Proc. ICASSP*, 2015, pp. 5206–5210.

[17] Yukun Zhu, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler, "Aligning books and movies: Towards story-like visual explanations by watching movies and reading books," in *Proc. ICCV*, 2015, pp. 19–27.

[18] Kazuki Irie, Rohit Prabhavalkar, Anjuli Kannan, Antoine Bruguier, David Rybach, and Patrick Nguyen, "On the choice of modeling unit for sequence-to-sequence speech recognition," in *Proc. INTERSPEECH*, 2019, pp. 3800–3804.

[19] Anmol Gulati, James Qin, Chung-Cheng Chiu, Niki Parmar, Yu Zhang, Jiahui Yu, Wei Han, Shibo Wang, Zhengdong Zhang, Yonghui Wu, and Ruoming Pang, "Conformer: Convolution-augmented transformer for speech recognition," in *Proc. INTERSPEECH*, 2020, pp. 5036–5040.

[20] Daniel S. Park, William Chan, Yu Zhang, Chung-Cheng Chiu, Barret Zoph, Ekin D. Cubuk, and Quoc V. Le, "SpecAugment: A simple data augmentation method for automatic speech recognition," in *Proc. INTERSPEECH*, 2019, pp. 2613–2617.

[21] "TensorFlow Hub: `bert_en_uncased_l-12_h-768_a-12`," `https://tfhub.dev/tensorflow/bert_en_uncased_L-12_H-768_A-12/4`, accessed on Sep. 27, 2021.