

BECTRA: TRANSDUCER-BASED END-TO-END ASR WITH BERT-ENHANCED ENCODER

Yosuke Higuchi¹, Tetsuji Ogawa¹, Tetsunori Kobayashi¹, Shinji Watanabe²

¹Waseda University, Japan ²Carnegie Mellon University, USA

ABSTRACT

We present **BERT-CTC-Transducer** (BECTRA), a novel end-to-end automatic speech recognition (E2E-ASR) model formulated by the transducer with a BERT-enhanced encoder. Integrating a large-scale pre-trained language model (LM) into E2E-ASR has been actively studied, aiming to utilize versatile linguistic knowledge for generating accurate text. One crucial factor that makes this integration challenging lies in the vocabulary mismatch; the vocabulary constructed for a pre-trained LM is generally too large for E2E-ASR training and is likely to have a mismatch against a target ASR domain. To overcome such an issue, we propose BECTRA, an extended version of our previous BERT-CTC, that realizes BERT-based E2E-ASR using a vocabulary of interest. BECTRA is a transducer-based model, which adopts BERT-CTC for its encoder and trains an ASR-specific decoder using a vocabulary suitable for a target task. With the combination of the transducer and BERT-CTC, we also propose a novel inference algorithm for taking advantage of both autoregressive and non-autoregressive decoding. Experimental results on several ASR tasks, varying in amounts of data, speaking styles, and languages, demonstrate that BECTRA outperforms BERT-CTC by effectively dealing with the vocabulary mismatch while exploiting BERT knowledge.

Index Terms— transducer, BERT, masked language model, pre-trained language model, end-to-end speech recognition

1. INTRODUCTION

In the field of natural language processing (NLP), language model (LM) pre-training has achieved remarkable success and become a dominant paradigm. With well-designed self-supervised objectives and a dramatic increase in model capacity, large-scale LMs [1, 2] are pre-trained on a vast amount of text-only data to acquire versatile linguistic knowledge [3]. Such pre-trained LMs (PLMs) provide rich representations for boosting the performance of downstream NLP tasks while alleviating the heavy requirement of supervised data.

Inspired by the great success in NLP, there has been an increasing interest in adopting a PLM for end-to-end automatic speech recognition (E2E-ASR). Several works have introduced PLMs to E2E-ASR via N-best hypothesis rescoring [4–8] and knowledge distillation [9–12]. More recently, others have attempted to fine-tune a PLM directly for E2E-ASR [13–17]. This enables a model to exploit the LM’s powerful representations during training and inference while requiring a complex mechanism for bridging the speech-text gap and careful adjustment for stabilizing the fine-tuning process.

BERT-CTC is another possibility we have explored for incorporating a pre-trained masked LM, i.e., BERT, into the formulation of connectionist temporal classification (CTC) [18]. BERT-CTC efficiently uses BERT *without fine-tuning* to explicitly condition CTC on context-aware linguistic information, alleviating the conditional independence in the output dependency. While BERT-CTC has shown promising results, its performance is still limited

due to a discrepancy in its output vocabulary. BERT-CTC predicts a sequence via a masked LM-based iterative refinement algorithm [19] (e.g., Table 1) with repeated updates on BERT embeddings. This BERT-based refinement forces the model to be trained on the BERT vocabulary, which is often too large for ASR training and is likely to have a mismatch against a target ASR domain. In fact, as shown in [18], the BERT-CTC performance becomes less significant when compared with standard CTC and transducer-based models trained on a vocabulary constructed from ASR training text. A similar issue has been reported in [20], which has mentioned that alphabetic languages (e.g., English) are prone to the vocabulary mismatch due to the difference in subword units.

In this work, we propose **BERT-CTC-Transducer** (BECTRA), an extension of BERT-CTC that is capable of handling the vocabulary mismatch and exploiting BERT knowledge for improving ASR performance. BECTRA formulates E2E-ASR based on the transducer [21], which comprises an encoder enhanced by BERT-CTC and a decoder (i.e., prediction/joint networks) trained with an ASR-specific vocabulary. Such a separate decoder enables a model to learn text generation more accurately in a suitable output unit, still benefiting from the BERT-conditioned framework. Moreover, BECTRA models E2E-ASR in a more accurate formulation than BERT-CTC, thanks to the transducer’s autoregressive characteristic relying less on conditional independence assumptions. With the combination of the transducer and BERT-CTC, we also propose a novel inference algorithm for BECTRA, which takes advantage of both autoregressive and non-autoregressive decoding algorithms.

2. BACKGROUND: BERT-CTC FOR END-TO-END ASR

Let $O = (\mathbf{o}_t \in \mathbb{R}^F | t = 1, \dots, T)$ be an input sequence of length T , and $W^b = (w_n^b \in \mathcal{V}^b | n = 1, \dots, N)$ be the corresponding output sequence of length N , where \mathbf{o}_t is an F -dimensional acoustic feature at frame t , w_n^b is an output token at position n , and \mathcal{V}^b is a vocabulary of BERT [1] trained on a target language. Note that the superscript (b) indicates the use of the BERT vocabulary. The objective of E2E-ASR is to formulate the direct mapping from O to W^b by modeling a posterior distribution $p(W^b | O)$ using a single deep neural network.

BERT-CTC [18] formulates $p(W^b | O)$ by introducing a masked sequence $\tilde{W}^b = (\tilde{w}_n^b \in \mathcal{V}^b \cup \{\emptyset\} | n = 1, \dots, N)$, which is obtained by replacing some tokens in an output sequence W^b with a special mask token \emptyset (see Table 1 for example). Considering all possible masking patterns compatible with W^b , $p(W^b | O)$ is factorized as

$$p(W^b | O) = \sum_{\tilde{W}^b} p(W^b | \tilde{W}^b, O) p(\tilde{W}^b | O). \quad (1)$$

Similar to CTC [22], $p(W^b | \tilde{W}^b, O)$ in Eq. (1) is marginalized over all possible alignment paths between O and W^b as

$$p(W^b | \tilde{W}^b, O) \approx \sum_{A^b \in \mathcal{B}^{-1}(W^b)} p(A^b | W^b, O) p(W^b | \tilde{W}^b), \quad (2)$$

where $A^b = (a_t^b \in \mathcal{V}^b \cup \{\epsilon\} | t = 1, \dots, T)$ is a frame-level token

sequence obtained by augmenting W^b with a blank symbol ϵ , and $\mathcal{B} : A^b \mapsto W^b$ is the collapsing function [22] that removes token repetitions and blank symbols from A^b . To obtain Eq. (2), BERT-CTC assumes reasonable conditional independence of \tilde{W}^b and O . The alignment probability $p(A^b|W^b, O)$ is further factorized using the probabilistic chain rule as

$$\text{Eq. (2)} \approx \sum_{A^b} \prod_{t=1}^T p(a_t^b | a_1^b, \dots, a_{t-1}^b, W^b, O) p(W^b | \tilde{W}^b), \quad (3)$$

which makes the same conditional independence assumption identical to CTC. Assuming $p(W^b|\tilde{W}^b)$ as a strong prior probability modeled by a pre-trained masked LM, BERT-CTC models the product of $p(a_t^b|\tilde{W}^b, O)$ and $p(W^b|\tilde{W}^b)$ in Eq. (3) as

$$\text{Eq. (3)} \triangleq \sum_{A^b} \prod_t p(a_t^b | \text{BERT}(\tilde{W}^b), O), \quad (4)$$

where $\text{BERT}(\cdot)$ indicates contextual embedding obtained from the output of BERT [1]. Different from standard CTC formulation, in Eq. (4), BERT-CTC conditions the token emission probability on BERT's contextual linguistic information. This enables a model to not only utilize BERT knowledge for E2E-ASR but also explicitly relax the conditional independence assumption made in CTC. For more detailed derivation from Eq. (2) to Eq. (4), see Sec. 3 of [18].

The blue region in Fig. 1 depicts the BERT-CTC architecture, consisting of an audio encoder, BERT, and a concatenation network. The token emission probability in Eq. (4) is computed as

$$p(a_t^b | \text{BERT}(\tilde{W}^b), O) = \sigma(\text{ConcatNet}_t(E, H)) \in [0, 1]^{|V^b|+1}, \quad (5)$$

$$E = \text{AudioEnc}(O) \in \mathbb{R}^{T \times D}, \quad H = \text{BERT}(\tilde{W}^b) \in \mathbb{R}^{N \times D}, \quad (6)$$

where D is the dimension of hidden vectors $\mathbf{e}_t \in E$ and $\mathbf{h}_n \in H$ in each encoded sequence, and $\sigma(\cdot)$ is a softmax layer. In Eq. (5), $\text{ConcatNet}_t(\cdot)$ indicates the t -th output of the concatenation network, which adopts the self-attention mechanism [23] for learning inner/inter dependencies within/between the concatenated E and H .

Training The objective function of BERT-CTC is defined by the negative log-likelihood of Eq. (1) expanded with Eq. (4):

$$\begin{aligned} \mathcal{L}_{\text{bec}} &= -\log \sum_{\tilde{W}^b} \sum_{A^b} \prod_t p(a_t^b | \text{BERT}(\tilde{W}^b), O) p(\tilde{W}^b | O) \\ &\leq -\mathbb{E}_{\tilde{W}^b \sim \mathcal{M}(W^b)} \left[\log \sum_{A^b} \prod_t p(a_t^b | \text{BERT}(\tilde{W}^b), O) \right], \quad (7) \end{aligned}$$

where the intractable marginalization over \tilde{W}^b is rewritten under expectation with respect to a sampling distribution $\mathcal{M}(W^b)$. Following [19], the sampling process is performed by first sampling the number of tokens from a uniform distribution as $M \sim \mathcal{U}(1, N)$ and then randomly masking M tokens in a ground-truth sequence W^b . The summation over A^b is efficiently computed as in CTC [22].

Inference Algorithm 1 represents a pseudocode for BERT-CTC inference, which is inspired by the mask-predict algorithm [19] combined with CTC inference [24–26]. The algorithm gradually generates an output sequence \hat{W}^b by iterating the following procedure K times. At each iteration $k \in \{1, \dots, K\}$, a masked sequence \hat{W}^b is fed into BERT to obtain contextual embeddings H (line 4). Given H and an encoder output E , \hat{W}^b is updated with a newly predicted sequence, which is obtained via the best path decoding [22] using the framewise token probability from Eq. (5) (line 5). Then, M tokens having the lowest probability scores are replaced with the mask token \emptyset (lines 6 and 7), where M is calculated from a linear decay function $\lfloor |\hat{W}^b| \cdot \frac{K-k}{K} \rfloor$ similar to [19].

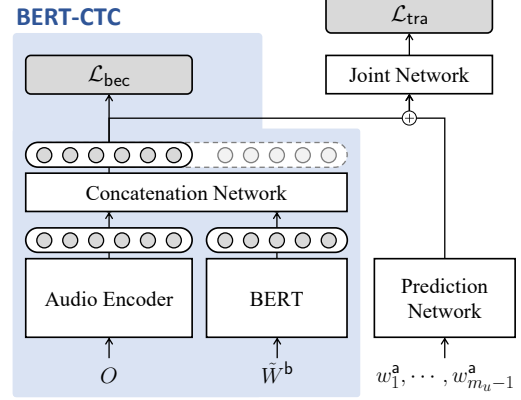


Fig. 1. Overview of BECTRA architecture. BECTRA adopts BERT-CTC as its encoder for transducer-based E2E-ASR training.

3. BECTRA

Overview Figure 1 illustrates the proposed model, namely **BERT-CTC-Transducer (BECTRA)**, that formulates E2E-ASR based on the transducer [21] and BERT-CTC. BECTRA constructs an encoder as the BERT-CTC model, where the output of the concatenation network in Eq. (5) is used to calculate the transducer loss. The joint and prediction networks are trained with an ASR-specific vocabulary, here a vocabulary constructed from the transcription of a target corpus. BECTRA enables more suitable training of E2E-ASR compared to BERT-CTC, unconstrained from the BERT vocabulary.

Let $W^a = (w_m^a \in \mathcal{V}^a | m = 1, \dots, M)$ be an M -length output sequence corresponding to an input sequence O and output sequence in the BERT vocabulary W^b , where \mathcal{V}^a is a vocabulary constructed from ASR training text. The superscript (a) indicates the use of the ASR vocabulary. Similar to Eq. (1), BECTRA formulates E2E-ASR by marginalizing the conditional probability $p(W^a|O)$ over all possible masked sequences as

$$p(W^a|O) = \sum_{\tilde{W}^b} p(W^a|\tilde{W}^b, O) p(\tilde{W}^b|O). \quad (8)$$

Note that, in Eq. (8), \tilde{W}^b is obtained by masking a sequence in the BERT unit W^b ($\neq W^a$). As in Eq. (2), $p(W^a|\tilde{W}^b, O)$ is factorized by considering possible alignment paths and making conditional independence assumptions as

$$p(W^a|\tilde{W}^b, O) \approx \sum_{A^a \in \mathcal{B}'^{-1}(W^a)} p(A^a|W^a, O) p(W^a|\tilde{W}^b), \quad (9)$$

where $A^a = (a_u^a \in \mathcal{V}^a \cup \{\epsilon\} | u = 1, \dots, T+M)$ is an alignment path between O and W^a , defined by the transducer [21], and $\mathcal{B}' : A^a \mapsto W^a$ is the collapsing function. The alignment probability $p(A^a|W^a, O)$ is further factorized by the probabilistic chain rule *without a conditional independence assumption* (cf. Eq. (3)) as

$$\text{Eq. (9)} = \sum_{A^a} \prod_{u=1}^{T+M} p(a_u^a | a_1^a, \dots, a_{u-1}^a, W^a, O) p(W^a|\tilde{W}^b), \quad (10)$$

$$\approx \sum_{A^a} \prod_{u=1}^{T+M} p(a_u^a | \underbrace{w_1^a, \dots, w_{m_u}^a}_{=\mathcal{B}'(a_1^a, \dots, a_{u-1}^a)}, W^a, O) p(W^a|\tilde{W}^b), \quad (11)$$

where m_u is the number of tokens predicted up to an index of u . In Eq. (11), we approximate $(a_1^a, \dots, a_{u-1}^a) \approx (w_1^a, \dots, w_{m_u}^a)$ as performed in the transducer, which is reasonable since W^a can be decided uniquely by the collapsing function. Similar to BERT-CTC

BERT-CTC ($k=1$)	... thou again meet any one after thee hour reciting artht of poetryry whether he be near'or far it will be a ...
BERT-CTC ($k=5$)	... thou again meet any one after this hour reciting artht of poetryry whether he be near or far it will be i ...
BERT-CTC ($k=10$)	... thou again meet any one after this hour reciting aught of poetryry whether he be near or far it will be i ...
BECTRA ($B=5$)	... thou again meet any one after this hour reciting aught of poetry whether he be near or far it will be i ...
Reference	... thou again meet any one after this hour reciting aught of poetry whether he be near or far it will be i ...

Table 1. Example inference process of BECTRA (Alg. 2), recognizing an utterance from LibriSpeech test-other set (2033-164914-0016). In BERT-CTC decoding (Alg. 2 line 2), the highlighted tokens are replaced with the mask token \emptyset and repredicted in the next iteration. The BERT-CTC result is further refined via the transducer decoding (Alg. 2 line 3). Blue indicates corrected tokens, and red indicates ones not.

Algorithm 1 Decoding algorithm of BERT-CTC

```

1: function DECODEBERTCTC( $E, K$ )
2:   Initialize a hypothesis  $\hat{W}^b$  with all mask tokens
3:   for  $k = 1, \dots, K$  do
4:     Calculate  $H = \text{BERT}(\hat{W}^b)$ 
5:     Update  $\hat{W}^b$  via the best path decoding based on Eq. (5)
6:     Calculate the number of masked tokens as  $M = \lfloor |\hat{W}^b| \cdot \frac{K-k}{K} \rfloor$ 
7:     Update  $\hat{W}^b$  by masking  $M$  tokens with the lowest probabilities
8:   return  $\hat{W}^b, H$ 

```

Algorithm 2 Decoding algorithm of BECTRA

```

1: function DECODEBECTRA( $E, K, B$ )
2:   Perform DECODEBERTCTC( $E, K$ ) and obtain  $H$ 
3:   Predict  $\hat{W}^a$  via the beam search decoding with a beam size of  $B$ ,
   using the joint and prediction networks in Eqs. (13) and (14)
4:   return  $\hat{W}^a$ 

```

in Eq. (4), BECTRA models Eq. (11) as

$$\text{Eq. (11)} \triangleq \sum_{A^a} \prod_u p(a_u^a | w_{< m_u}^a, \text{BERT}(\tilde{W}^b), O), \quad (12)$$

where we assume $p(W^a | \tilde{W}^b)$ can be modeled by BERT, as W^b and W^a are convertible to each other via their tokenizers.

The architecture of BECTRA is shown in Fig. 1, and the token emission probability in Eq. (12) is computed as

$$p(a_u^a | w_{< m_u}^a, \text{BERT}(\tilde{W}^b), O) = \sigma(\text{JointNet}(\text{ConcatNet}_t(E, H), \mathbf{q}_{m_u})) \in [0, 1]^{|\mathcal{V}^a|+1}, \quad (13)$$

$$\mathbf{q}_{m_u} = \text{PredictionNet}(w_1^a, \dots, w_{m_u-1}^a) \in \mathbb{R}^D, \quad (14)$$

where \mathbf{q}_{m_u} is a D -dimensional hidden vector obtained by encoding the previous non-blank tokens using the prediction network. In Eq. (13), the joint network maps the combined outputs of the concatenation and prediction networks into a joint space. The adoption of the prediction network allows the model to explicitly capture causal dependency in output tokens, which is one of the key differences BECTRA has an advantage over BERT-CTC.

Training By substituting Eq. (12) into Eq. (8), the transducer loss of BECTRA is derived in the same manner as Eq. (7) as

$$\mathcal{L}_{\text{tra}} \triangleq -\mathbb{E}_{\tilde{W}^b \sim \mathcal{M}(W^b)} \left[\log \sum_{A^a} \prod_u p(a_u^a | w_{< m_u}^a, \text{BERT}(\tilde{W}^b), O) \right], \quad (15)$$

where the summation over A^a is efficiently computed via dynamic programming [21]. The objective function of BECTRA is defined by combining \mathcal{L}_{bec} from Eq. (7) and \mathcal{L}_{tra} from Eq. (15) as

$$\mathcal{L}_{\text{bectra}} = (1 - \lambda) \mathcal{L}_{\text{bec}} + \lambda \mathcal{L}_{\text{tra}}, \quad (16)$$

where $\lambda \in (0, 1)$ is a tunable parameter.

Inference Algorithm 2 shows the inference algorithm of BECTRA, implemented with BERT-CTC decoding followed by beam-search

decoding of the transducer [21, 27] (see Table 1 for example). BERT-CTC decoding provides the model with a fully contextualized BERT output H , which is from the final hypothesis estimated by the iterative refinement (line 2). The beam-search decoding uses the token emission probability from Eq. (13) to find an optimal sequence with the highest sequence-level generation probability (line 3). With this combined inference algorithm, BECTRA can utilize BERT to capture bi-directional context in an output sequence, the advantage of non-autoregressive decoding. Moreover, the transducer-based decoding enables the model to further refine a sequence in an autoregressive manner, using a more suitable output unit for ASR.

4. EXPERIMENTS

4.1. Experimental Setting

We used the ESPnet toolkit [28] for conducting the experiments, and all the codes and recipes are made publicly available at <https://github.com/YosukeHiguchi/espnet/tree/bectra>.

Data We evaluated models using various corpora with different amounts of data, speaking styles, and languages, including LibriSpeech (LS) [29], TED-LIUM2 (TED2) [30], and AISHELL-1 (AS1) [31]. We also trained models on the *train-clean-100* subset of LS (LS-100) for performing additional investigations and analyses. To obtain the vocabulary \mathcal{V}^a from ASR transcriptions, we used SentencePiece [32] to construct subword vocabularies for LS-100, LS and TED2, where each vocabulary size $|\mathcal{V}^a|$ was set to 300, 5k, and 500, respectively. For AS1, we used character-level tokenization with 4231 Chinese characters.

Evaluated models For a baseline model, we trained Conformer-Transducer (Cfm-T) implemented in ESPnet [27], which is similar to the model in Fig. 1 but without the concatenation, CTC, and BERT components. BERT-CTC [18] is also a baseline model trained based on \mathcal{L}_{bec} from Eq. (7). BECTRA is the proposed model trained based on $\mathcal{L}_{\text{bectra}}$ from Eq. (16).

Network architecture For the audio encoder, we constructed the Conformer architecture [33], which consisted of two convolutional neural network layers followed by a stack of 12 encoder blocks. The number of heads d_h , dimension of a self-attention layer d_{model} , dimension of a feed-forward network d_{ff} , and kernel size K were set to 4, 256, 1024, and 31, respectively. For the transducer-based models, the prediction network was a single long short-term memory layer with 256 hidden units. For the models using BERT, the concatenation network was the Transformer encoder [23] with 6 blocks, and d_h , d_{model} , and d_{ff} were set to 4, 256, and 2048, respectively. We used a BERT_{BASE} of each language provided in HuggingFace [34]: English [35] ($|\mathcal{V}^b| = 30522$) and Mandarin [36] ($|\mathcal{V}^b| = 21128$). The output dimension D of the audio encoder \mathbf{e}_t , BERT \mathbf{h}_n , and prediction network \mathbf{q}_{m_u} were all adjusted to match $d_{\text{model}} (= 256)$.

Training and decoding We mostly followed configurations provided by the ESPnet2 recipe for each dataset. The models were trained to 100 epochs for LS-100 and AS1, and 70 epochs for TED2

Table 2. Word or character error rates [%] (\downarrow) of our proposed BECTRA compared to Conformer-Transducer (Cfm-T) and BERT-CTC baselines. Each model was trained using either a vocabulary used in BERT (\mathcal{V}^b) or vocabulary constructed from ASR training text (\mathcal{V}^a).

Model	Output Vocab.	LibriSpeech-100h				LibriSpeech-960h				TED-LIUM2		AISHELL-1	
		Dev WER		Test WER		Dev WER		Test WER		Dev WER	Test WER	Dev CER	Test CER
		clean	other	clean	other	clean	other	clean	other				
Cfm-T	\mathcal{V}^b	9.7	21.5	9.8	22.3	—	—	—	—	—	—	—	—
Cfm-T	\mathcal{V}^a	5.9	17.7	6.0	17.6	2.5	6.8	2.8	6.8	7.8	7.4	4.9	5.3
BERT-CTC	\mathcal{V}^b	7.0	16.4	7.1	16.5	3.1	7.1	3.2	7.1	8.3	7.6	3.9	4.0
BECTRA	\mathcal{V}^a	5.1	15.4	5.4	15.5	2.6	6.7	2.9	6.7	7.3	6.9	3.7	3.9

and LS. The Adam optimizer [37] with Noam learning rate scheduling [23] was used for weight updates, where warmup steps and a peak learning rate were set to 15k and 2e-3, respectively. We augmented speech data using speed perturbation [38] with a factor of 3 and SpecAugment [39]. Similar to [40], we adopted the intermediate CTC regularization [41] for the audio encoder, where an auxiliary CTC loss was applied to the 6-th layer and calculated based on the ASR vocabulary \mathcal{V}^a . For BECTRA training, we set λ to 0.5 in Eq. (16). After training, a final model was obtained for evaluation by averaging model parameters over ten checkpoints with the best validation performance. For the number of iterations in BERT-CTC decoding (in Alg. 1), we set K to 20 for BERT-CTC and 10 for BECTRA. We performed the beam search decoding with a beam size of 10 for Cfm-T and 5 for BECTRA.

4.2. Difficulty of Training ASR with BERT Vocabulary

In Table 2, we compare LS-100 results on Cfm-T trained with the the BERT or ASR vocabulary (\mathcal{V}^b vs. \mathcal{V}^a). At a glance, Cfm-T resulted in significantly worse WERs by using the BERT vocabulary, suggesting unsuitable ASR training with the word-level and domain-mismatched BERT units. We also mention that the large vocabulary size leads to increasingly high memory consumption during the transducer training [40], which makes it difficult to train a model, especially on large datasets. In contrast, BERT-CTC in Table 2 achieved decent results by effectively dealing with the vocabulary mismatch, explicitly using BERT information [18].

4.3. Main Results

Table 2 lists the main results of all the tasks evaluated in the word error rate (WER) or character error rate (CER). Comparing Cfm-T (with \mathcal{V}^a) and BERT-CTC, BERT-CTC performed worse than Cfm-T in several tasks due to the vocabulary discrepancy. For example, in TED2, a severe domain mismatch exists between the BERT training text and ASR transcription (i.e., Wikipedia vs. lecture) that caused unsuitable ASR training for BERT-CTC. Overall, BECTRA achieved the best result across all of the tasks. The gain from Cfm-T demonstrates the effectiveness of exploiting BERT knowledge via the BERT-CTC-based encoder. Moreover, BECTRA improved BERT-CTC by generating text in a suitable output unit and mitigating the conditional independence assumption (Eq. (3) vs. Eq. (11)). Another notable observation was that, with more data in LS-960, the performance gap between Cfm-T and BECTRA was reduced, and the usage of BERT became less influential. We assume that LS-960 contained enough text data that the models were able to learn rich linguistic information in the LibriSpeech-specific domain.

Table 1 shows an example decoding process of BECTRA. BERT-CTC succeeded at resolving most of the substitution errors via iterative refinement using BERT. The transducer decoding was particularly effective at adjusting deletion and insertion errors in BERT-CTC outputs, e.g., “poetryry” \rightarrow “poetry” in Table 1.

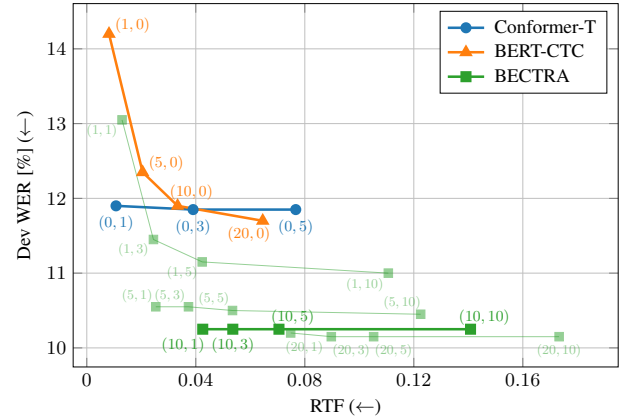


Fig. 2. Trade-off between WER and RTF on LS dev. sets. The values in parenthesis show the number of iterations and a beam size (K, B).

4.4. Trade-off between WER and Inference Speed

Figure 2 depicts the trade-off between WER and real-time factor (RTF) on the LS dev.-{clean, other} sets. RTF was measured using a single V100 GPU with a batchsize of 1. Looking at the results from greedy decoding ($K \leq 1$ and $B = 1$), Cfm-T resulted in the best performance with the lowest WER and RTF. The performance gain was more substantial for increasing the number of iterations in BERT-CTC than for increasing a beam size in Cfm-T. BECTRA greatly benefited from beam search decoding at $K = 1$, but the improvement became small with more iterations $K > 1$. This indicates that BERT-CTC decoding refined the output sequence enough that the search space was reduced during beam search decoding, and only a small beam size ($K = 3$ or 5) is adequate without degrading the inference speed. All in all, BECTRA with $K = 10$ and $B \in [1, 5]$ resulted in the preferable performance balancing the trade-off reasonably well, taking advantage of both fast non-autoregressive decoding of BERT-CTC and accurate autoregressive decoding of Cfm-T.

5. CONCLUSION

We proposed BECTRA, a novel E2E-ASR model formulated by the transducer and BERT-CTC. BECTRA adopts BERT-CTC for the audio encoder, aiming to exploit BERT for capturing contextual information. In addition, prediction and joint networks are trained with an ASR-specific vocabulary, which enables the model to perform ASR in a suitable unit. One of the limitations of BECTRA is its non-streaming property due to the BERT-CTC’s non-autoregressive formulation. The possible solutions include the adoption of the two-pass modeling [42] or blockwise-attention mechanism [43].

Acknowledgement This work was supported in part by JST ACT-X (JPMJAX210J) and JSPS KAKENHI (JP21J23495).

6. REFERENCES

- [1] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional Transformers for language understanding," in *Proc. NAACL-HLT*, 2019, pp. 4171–4186.
- [2] T. Brown, B. Mann, N. Ryder, M. Subbiah *et al.*, "Language models are few-shot learners," in *Proc. NeurIPS*, 2020, pp. 1877–1901.
- [3] I. Tenney, D. Das, and E. Pavlick, "BERT rediscovers the classical NLP pipeline," in *Proc. ACL*, 2019, pp. 4593–4601.
- [4] J. Shin, Y. Lee, and K. Jung, "Effective sentence scoring method using BERT for speech recognition," in *Proc. ACML*, 2019, pp. 1081–1093.
- [5] J. Salazar, D. Liang, T. Q. Nguyen, and K. Kirchhoff, "Masked language model scoring," in *Proc. ACL*, 2020, pp. 2699–2712.
- [6] S.-H. Chiu and B. Chen, "Innovative BERT-based reranking language models for speech recognition," in *Proc. SLT*, 2021, pp. 266–271.
- [7] H. Futami, H. Inaguma, M. Mimura, S. Sakai *et al.*, "ASR rescoring and confidence estimation with ELECTRA," in *Proc. ASRU*, 2021, pp. 380–387.
- [8] T. Udagawa, M. Suzuki, G. Kurata, N. Itoh *et al.*, "Effect and analysis of large-scale language model rescoring on competitive ASR systems," in *Proc. Interspeech*, 2022, pp. 3919–3923.
- [9] H. Futami, H. Inaguma, S. Ueno, M. Mimura *et al.*, "Distilling the knowledge of BERT for sequence-to-sequence ASR," in *Proc. Interspeech*, 2020, pp. 3635–3639.
- [10] Y. Bai, J. Yi, J. Tao, Z. Tian *et al.*, "Fast end-to-end speech recognition via non-autoregressive models and cross-modal knowledge transferring from BERT," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 29, pp. 1897–1911, 2021.
- [11] Y. Kubo, S. Karita, and M. Bacchiani, "Knowledge transfer from large-scale pretrained language models to end-to-end speech recognizers," in *Proc. ICASSP*, 2022, pp. 8512–8516.
- [12] K.-H. Lu and K.-Y. Chen, "A context-aware knowledge transferring strategy for CTC-based ASR," in *Proc. SLT*, 2022.
- [13] W.-C. Huang, C.-H. Wu, S.-B. Luo, K.-Y. Chen *et al.*, "Speech recognition by simply fine-tuning BERT," in *Proc. ICASSP*, 2021, pp. 7343–7347.
- [14] C. Yi, S. Zhou, and B. Xu, "Efficiently fusing pretrained acoustic and linguistic encoders for low-resource speech recognition," *IEEE Signal Process. Lett.*, vol. 28, pp. 788–792, 2021.
- [15] G. Zheng, Y. Xiao, K. Gong, P. Zhou *et al.*, "Wav-BERT: Co-operative acoustic and linguistic representation learning for low-resource speech recognition," in *Proc. Findings of EMNLP*, 2021, pp. 2765–2777.
- [16] K. Deng, S. Cao, Y. Zhang, and L. Ma, "Improving hybrid CTC/attention end-to-end speech recognition with pretrained acoustic and language models," in *Proc. ASRU*, 2021, pp. 76–82.
- [17] F.-H. Yu, K.-Y. Chen, and K.-H. Lu, "Non-autoregressive ASR modeling using pre-trained language models for Chinese speech recognition," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 30, pp. 1474–1482, 2022.
- [18] Y. Higuchi, B. Yan, S. Arora, T. Ogawa *et al.*, "BERT meets CTC: New formulation of end-to-end speech recognition with pre-trained masked language model," in *Proc. Findings of EMNLP*, 2022.
- [19] M. Ghazvininejad, O. Levy, Y. Liu, and L. Zettlemoyer, "Mask-predict: Parallel decoding of conditional masked language models," in *Proc. EMNLP-IJCNLP*, 2019, pp. 6114–6123.
- [20] K. Deng, Z. Yang, S. Watanabe, Y. Higuchi *et al.*, "Improving non-autoregressive end-to-end speech recognition with pre-trained acoustic and language models," in *Proc. ICASSP*, 2022, pp. 8522–8526.
- [21] A. Graves, "Sequence transduction with recurrent neural networks," *arXiv preprint arXiv:1211.3711*, 2012.
- [22] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks," in *Proc. ICML*, 2006, pp. 369–376.
- [23] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit *et al.*, "Attention is all you need," in *Proc. NeurIPS*, 2017, pp. 5998–6008.
- [24] W. Chan, C. Saharia, G. Hinton, M. Norouzi *et al.*, "Imputer: Sequence modelling via imputation and dynamic programming," in *Proc. ICML*, 2020, pp. 1403–1413.
- [25] Y. Higuchi, S. Watanabe, N. Chen, T. Ogawa *et al.*, "Mask CTC: Non-autoregressive end-to-end ASR with CTC and mask predict," in *Proc. Interspeech*, 2020, pp. 3655–3659.
- [26] Y. Higuchi, H. Inaguma, S. Watanabe, T. Ogawa *et al.*, "Improved mask-CTC for non-autoregressive end-to-end ASR," in *Proc. ICASSP*, 2021, pp. 8363–8367.
- [27] F. Boyer, Y. Shinohara, T. Ishii, H. Inaguma *et al.*, "A study of Transducer based end-to-end ASR with ESPnet: Architecture, auxiliary loss and decoding strategies," in *Proc. ASRU*, 2021, pp. 16–23.
- [28] S. Watanabe, T. Hori, S. Karita, T. Hayashi *et al.*, "ESPnet: End-to-end speech processing toolkit," in *Proc. Interspeech*, 2018, pp. 2207–2211.
- [29] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: An ASR corpus based on public domain audio books," in *Proc. ICASSP*, 2015, pp. 5206–5210.
- [30] A. Rousseau, P. Deléglise, and Y. Estève, "Enhancing the TED-LIUM corpus with selected data for language modeling and more TED talks," in *Proc. LREC*, 2014, pp. 3935–3939.
- [31] H. Bu, J. Du, X. Na, B. Wu *et al.*, "AISHELL-1: An open-source Mandarin speech corpus and a speech recognition baseline," in *Proc. O-COCOSDA*, 2017, pp. 1–5.
- [32] T. Kudo, "Subword regularization: Improving neural network translation models with multiple subword candidates," in *Proc. ACL*, 2018, pp. 66–75.
- [33] A. Gulati, J. Qin, C.-C. Chiu, N. Parmar *et al.*, "Conformer: Convolution-augmented Transformer for speech recognition," in *Proc. Interspeech*, 2020, pp. 5036–5040.
- [34] T. Wolf, L. Debut, V. Sanh, J. Chaumond *et al.*, "Transformers: State-of-the-art natural language processing," in *Proc. EMNLP: System Demonstrations*, 2020, pp. 38–45.
- [35] "bert-base-uncased," <https://huggingface.co/bert-base-uncased>, [Online; Accessed on October-10-2022].
- [36] "bert-base-chinese," <https://huggingface.co/bert-base-chinese>, [Online; Accessed on October-10-2022].
- [37] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. ICLR*, 2015.
- [38] T. Ko, V. Peddinti, D. Povey, and S. Khudanpur, "Audio augmentation for speech recognition," in *Proc. Interspeech*, 2015, pp. 3586–3589.
- [39] D. S. Park, W. Chan, Y. Zhang, C.-C. Chiu *et al.*, "SpecAugment: A simple data augmentation method for automatic speech recognition," in *Proc. Interspeech*, 2019, pp. 2613–2617.
- [40] J. Lee, L. Lee, and S. Watanabe, "Memory-efficient training of RNN-Transducer with sampled softmax," in *Proc. Interspeech*, 2022.
- [41] J. Lee and S. Watanabe, "Intermediate loss regularization for CTC-based speech recognition," in *Proc. ICASSP*, 2021, pp. 6224–6228.
- [42] T. N. Sainath, R. Pang, D. Rybach, Y. He *et al.*, "Two-pass end-to-end speech recognition," in *Proc. Interspeech*, 2019, pp. 2773–2777.
- [43] T. Wang, Y. Fujita, X. Chang, and S. Watanabe, "Streaming end-to-end ASR based on blockwise non-autoregressive models," in *Proc. Interspeech*, 2021, pp. 3755–3759.