ASR RESCORING AND CONFIDENCE ESTIMATION WITH ELECTRA

Hayato Futami, Hirofumi Inaguma, Masato Mimura, Shinsuke Sakai, Tatsuya Kawahara

Graduate School of Informatics, Kyoto University, Sakyo-ku, Kyoto, Japan

ABSTRACT

In automatic speech recognition (ASR) rescoring, the hypothesis with the fewest errors should be selected from the n-best list using a language model (LM). However, LMs are usually trained to maximize the likelihood of correct word sequences, not to detect ASR errors. We propose an ASR rescoring method for directly detecting errors with ELECTRA, which is originally a pre-training method for NLP tasks. ELECTRA is pre-trained to predict whether each word is replaced by BERT or not, which can simulate ASR error detection on large text corpora. To make this pre-training closer to ASR error detection, we further propose an extended version of ELECTRA called phone-attentive ELECTRA (P-ELECTRA). In the pre-training of P-ELECTRA, each word is replaced by a phone-to-word conversion model, which leverages phone information to generate acoustically similar words. Since our rescoring method is optimized for detecting errors, it can also be used for word-level confidence estimation. Experimental evaluations on the Librispeech and TED-LIUM2 corpora show that our rescoring method with ELECTRA is competitive with conventional rescoring methods with faster inference. ELECTRA also performs better in confidence estimation than BERT because it can learn to detect inappropriate words not only in fine-tuning but also in pre-training.

Index Terms— speech recognition, language model, ELECTRA, rescoring, confidence estimation

1. INTRODUCTION

Automatic speech recognition (ASR) has been realized by DNN-HMM hybrid systems with an acoustic model (AM) and a language model (LM). Recently, end-to-end ASR that integrates an AM and an LM into a single neural network has achieved prominent performances. There are some architectures for end-to-end modeling: connectionist temporal classification (CTC) [1], attention-based encoder-decoder models [2, 3, 4], and neural network transducer models [5, 6].

End-to-end ASR models are trained on paired speech and transcripts. While the amount of paired data for the target domain is limited, a large amount of text-only data is often available. An external LM trained on text-only data is usually applied to improve ASR performance. Shallow Fusion [7, 8, 9] and n-best rescoring are widely used for applying an external LM to end-to-end ASR. In Shallow Fusion, the interpolated score of the LM and the ASR model is used in beam search decoding. In n-best rescoring, which we focus on in this study, an n-best list obtained from the ASR model is scored using the LM, then the hypothesis of the highest interpolated score is selected.

RNN and Transformer LMs are conventionally used for rescoring. They predict each word on the basis of its preceding context in an autoregressive manner. These autoregressive LMs are usually trained on sentences without errors by maximizing the likelihood of the word sequences. In rescoring, however, LMs should discriminate a hypothesis with fewer ASR errors from other hypotheses with more errors. Several studies have pointed out that the training objective of LMs is sub-optimal for rescoring, and they proposed discriminative LMs [10, 11, 12], which are trained with discriminative criteria using ASR results and their corresponding references. They are obtained from paired speech and transcripts, the scale of which is usually limited compared with text-only data.

Recently, BERT [13] has been used for rescoring [14, 15]. BERT was originally proposed as a pre-training method for NLP tasks such as question answering and language understanding. BERT can also be regarded as an external LM that predicts each masked word on the basis of both its preceding and following context. BERT performs better in rescoring than conventional autoregressive LMs because of the deeply bidirectional architecture of its Transformer encoder [14, 15]. However, rescoring with BERT is too time-consuming. It takes L inference steps to rescore a hypothesis of length L [15] by masking each word iteratively, while Transformer LM takes a single step [16].

In this study, we propose an ASR rescoring method for detecting then counting ASR errors with ELECTRA [17], which is a pre-training method with a deeply bidirectional architecture like BERT. Different from BERT, ELECTRA is pre-trained for a replaced token detection task instead of masked language model (MLM), in which the generator (BERT) replaces some words of the input by sampling and the discriminator is trained to predict whether each word is replaced by BERT or not. ASR error detection can be trained only on paired data, but ELECTRA can simulate it on large text corpora. However, there is a mismatch between real ASR errors and BERT's replacement used in the pre-training of ELECTRA because ASR takes acoustic features as input, while BERT does not. To solve the mismatch, we further investigate two methods of making ELECTRA's training conditions closer to rescoring conditions: fine-tuning ELECTRA on ASR results for error detection and introducing phone-attentive ELECTRA (P-ELECTRA). P-ELECTRA is a modified version of ELECTRA, which employs a phone-to-word conversion model as a generator instead of BERT. With the help of phone information, we can obtain replacements that are similar to ASR errors on text corpora.

Our rescoring method with ELECTRA can solve the two issues mentioned above: the mismatch between LM training and rescoring objectives and slow inference in rescoring with bidirectional architecture. First, it is optimized directly to the rescoring objective by detecting errors in each hypothesis and selecting the hypothesis with the fewest errors in the n-best list. Second, it can benefit from deeply bidirectional contextual information with faster inference than BERT, as it takes only a single step to rescore without masking. Moreover, rescoring with ELECTRA is faster than with conventional LMs. ELECTRA conducts binary classification for rescoring, while conventional LMs conduct word prediction, the complexity of which is proportional to the vocabulary size.

The proposed rescoring method is closely related to confidence estimation, or the ASR error detection task. Confidence estimation assesses the quality of ASR predictions [18, 19, 20, 21, 22, 23, 24, 25], which is useful for many downstream ASR applications such as voice assistants. We demonstrate that our models for rescoring can be applied to confidence estimation without any additional architectural changes or training. ELECTRA is pre-trained for the replaced token detection task that is close to confidence estimation, and therefore it can effectively leverage text-only data.

2. PRELIMINARIES AND RELATED WORK

2.1. ELECTRA

ELECTRA [17] is a pre-training method for downstream NLP tasks like BERT [13]. In BERT, masked language modeling (MLM) replaces some input tokens with `[MASK]`, then it is trained to predict the original tokens. ELECTRA instead employs a replaced token detection task for pre-training. It corrupts the input by replacing some tokens by sampling from a generator, then a discriminator is trained to predict whether each token is the original or replacement. While BERT learns from a small masked-out subset (usually 15%), ELECTRA can learn from all input tokens, which is more computationally efficient [17].

In the pre-training of ELECTRA, two Transformer-based models (generator G and discriminator D) are jointly trained. The procedure is formulated as follows. First, some tokens of the input $\boldsymbol{y} = (y_1, y_2, ..., y_L)$ are selected and masked out. Let $\boldsymbol{m} = (m_1, m_2, ..., m_k)(1 \leq m_i \leq L)$ denote the positions of masked-out tokens.

$$\boldsymbol{y}^{\text{masked}} = \text{replace}(\boldsymbol{y}, \boldsymbol{m}, \texttt{[MASK]}) \tag{1}$$

The generator G then predicts tokens of the masked positions and generates a corrupted example $\boldsymbol{y}^{\text{corrupt}}$ by sampling.

$$\begin{aligned} \boldsymbol{y}^{\text{corrupt}} &= \text{replace}(\boldsymbol{y}^{\text{masked}}, \boldsymbol{m}, \hat{\boldsymbol{y}}) \\ \hat{y}_i &\sim p_G(y_i|\boldsymbol{y}^{\text{masked}}), i \in \boldsymbol{m} \end{aligned} \tag{2}$$

The generator is BERT trained with the MLM objective, and its loss function is

$$\mathcal{L}_G = \sum_{i \in \boldsymbol{m}} -\log p_G(y_i|\boldsymbol{y}^{\text{masked}}) \tag{3}$$

The discriminator D is a binary classifier trained to discriminate the original token from the token replaced by the generator G. Let $D^{(i)}(\boldsymbol{y})$ denote the discriminator output passed through the sigmoid layer for the i-th token, which should be 1 when y_i is replaced. Its loss function is

$$\begin{aligned} \mathcal{L}_D = \sum_{i=1}^{L} &-\delta(y_i^{\text{corrupt}}, y_i)\log(1 - D^{(i)}(\boldsymbol{y}^{\text{corrupt}})) \\ &-(1 - \delta(y_i^{\text{corrupt}}, y_i))\log D^{(i)}(\boldsymbol{y}^{\text{corrupt}}) \end{aligned} \tag{4}$$

where $\delta(y_i^{\text{corrupt}}, y_i)$ becomes 1 when $y_i^{\text{corrupt}} = y_i$, and 0 otherwise. The weighted sum of $\mathcal{L}_G$ and $\mathcal{L}_D$ is minimized during pre-training. After pre-training, the discriminator is fine-tuned on downstream NLP tasks, and it is referred to as "ELECTRA". In this study, we use the discriminator for rescoring and confidence estimation tasks.

2.2. Rescoring

Rescoring is a simple and widely used method to apply LMs to end-to-end ASR models. An n-best list is generated by beam search with the ASR model, then each hypothesis in the list is rescored using the LM.

$$\mathit{Score}(\boldsymbol{X}, \boldsymbol{y}) = \log p_{\text{ASR}}(\boldsymbol{y}|\boldsymbol{X}) + \alpha \mathit{Score}_{\text{LM}}(\boldsymbol{y}) + \beta|\boldsymbol{y}| \tag{5}$$

where $\boldsymbol{X}$ denotes acoustic features, and $\boldsymbol{y} = (y_1, ..., y_L)$ denotes a hypothesis. The hypothesis that has the highest $\mathit{Score}(\boldsymbol{X}, \boldsymbol{y})$ is selected. As $\log p_{\text{ASR}}$ and $\mathit{Score}_{\text{LM}}$ tend to assign a higher score to a shorter hypothesis, $\beta|\boldsymbol{y}|$ encourages longer hypotheses to reduce deletion errors. Here, α and β are hyperparameters.

The $\mathit{Score}_{\text{LM}}$ is conventionally calculated using autoregressive LMs such as RNN and Transformer LMs. They provide a likelihood score for each hypothesis $\boldsymbol{y}$ as

$$\mathit{Score}_{\text{LM}}(\boldsymbol{y}) = \sum_{i=1}^{L} \log p(y_i|\boldsymbol{y}_{<i}) = \log p(y_1, ..., y_L) \tag{6}$$

where $\boldsymbol{y}_{<i} = (y_1, ..., y_{i-1})$. Autoregressive LMs predict each token using its preceding context. Transformer LM can calculate $p(y_i|\boldsymbol{y}_{<i})$ for all i in parallel with self-attention mechanism [16].

Recently, BERT has also been used for calculating $\mathit{Score}_{\text{LM}}$. It provides a pseudo-likelihood score [26] as

$$\mathit{Score}_{\text{LM}}(\boldsymbol{y}) = \sum_{i=1}^{L} \log p(y_i|\boldsymbol{y}_{\backslash i}) \tag{7}$$

where $\boldsymbol{y}_{\backslash i} = (y_1, ..., y_{i-1}, \texttt{[MASK]}, y_{i+1}, ..., y_L)$ (the i-th token of $\boldsymbol{y}$ is masked out). BERT is reported to perform better in rescoring than autoregressive LMs by predicting each token using both its preceding and following context [14, 15]. It also performs better than bidirectional RNN LMs [27, 28], because it is "deeply bidirectional" [13], while the bidirectional RNN is the shallow concatenation of two directions of RNNs [29]. However, rescoring with BERT takes L inference steps for a hypothesis of length L. It requires a iterative procedure to calculate $p(y_i|\boldsymbol{y}_{\backslash i})$ for each position i using different masked inputs $\boldsymbol{y}_{\backslash i}$. More recently, Electric [30] was proposed to calculate $p(y_i|\boldsymbol{y}_{\backslash i})$ for all i in a single step. It efficiently provides $\mathit{Score}_{\text{LM}}$ based on pseudo-likelihood through noise contrastive estimation training. Electric only learns the training data distribution, while ELECTRA learns whether each token comes from the data distribution or noise distribution by the generator in pre-training [30]. ELECTRA provides $\mathit{Score}_{\text{LM}}$ based on error detection. It can be fine-tuned on ASR hypotheses and directly applied to confidence estimation.

Another stream of ASR rescoring includes discriminative language modeling. LMs are usually trained to maximize the likelihood of word sequences without errors. However, this does not necessarily maximize the performance of ASR rescoring, in which the LM should discriminate the hypothesis with the fewest errors from hypotheses that contain more errors. In [10], the word-level log-likelihood ratio of ASR hypotheses and references was used as a training criterion for RNN LMs. In [11, 31], a minimum word error rate (MWER) training for RNN LMs was proposed. In [12, 32], a large margin criterion that enlarges the margin between hypotheses and references was applied to RNN LM, Transformer LM, and BERT training. In these studies, the likelihood of each word or sentence was adapted to the discriminative criterion using ASR hypotheses and references. In this study, we use ELECTRA to learn error detection that is in nature discriminative and can be trained not only on paired data but also on large text corpora.

2.3. Confidence estimation

Confidence estimation is an important task for ASR applications to mitigate the adverse effects of ASR errors, which are inevitable. For example, in voice assistants, queries of low confidence will be asked back. In audio transcription tasks, it helps manual corrections by flagging less confident words. Confidence estimation is also used in semi-supervised learning [33], in which utterances with confident predictions are selected as training data.

Confidence scores are estimated at word-level or utterance-level. In this study, we focus on word-level confidence estimation. In DNN-HMM ASR, reliable word-level confidence scores can be obtained from word posterior probabilities over lattices [18], and they are further improved by confidence estimation modules (CEMs) [19, 20, 21]. In seq2seq ASR models, confidence scores can be obtained from softmax probabilities of their decoders, but they are not reliable enough because of overconfidence [34]. In [22], a lightweight CEM that uses internal features of a seq2seq model was proposed to mitigate overconfidence. In [23], softmax temperature values for each token were predicted to adjust overconfident probabilities. In CTC-based ASR models we used in this study, confidence scores can be obtained with the forward-backward algorithm [35], which was reported to perform well [24]. End-to-end ASR models and CEMs usually use subword units such as byte pair encoding (BPE) [36] to handle rare and unknown words. They provide subword-level confidence scores, but in many downstream applications, word-level confidence scores are useful. Word-level confidence scores can be obtained simply by taking the average [22], product, or minimum [25] of subword-level scores if a word consists of multiple subwords. In [37], a subword-level CEM was directly trained with the word-level confidence objective. In these studies, CEMs were mainly trained with ASR results and their references obtained from paired data. In this study, ELECTRA is used as a CEM, which can be pre-trained by simulating the confidence estimation task on large text corpora.

3. PROPOSED METHOD

3.1. Rescoring with ELECTRA

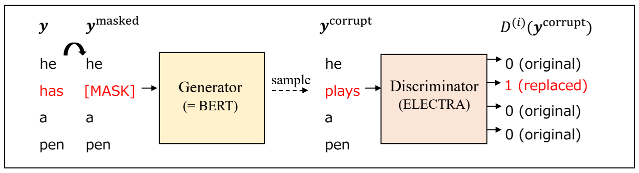
In ELECTRA, $D^{(i)}(\boldsymbol{y})$ is pre-trained to output 1 when the i-th token is replaced, and 0 otherwise, as illustrated in Fig. 1 (a). ELECTRA learns to detect syntactically or semantically inappropriate tokens, which is useful for detecting ASR errors. By counting the expected number of errors in a hypothesis, $Score_{\mathrm{LM}}(\boldsymbol{y})$ is defined as

$$Score_{\mathrm{LM}}(\boldsymbol{y}) = -\sum_{i=1}^{L} D^{(i)}(\boldsymbol{y}) \tag{8}$$

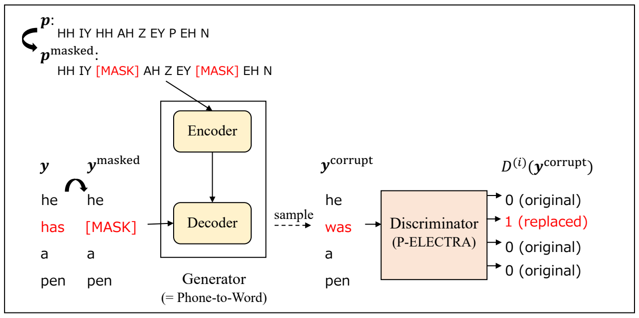
which is illustrated in Fig. 1 (c). Note that the score should be higher for hypotheses with fewer errors. Rescoring with ELECTRA can benefit from deeply bidirectional contextual information with a single-step inference per hypothesis, while BERT requires L steps. ELECTRA can look at all the tokens including y_i to calculate $D^{(i)}(\boldsymbol{y})$, and therefore it provides $D^{(i)}(\boldsymbol{y})$ for all i in parallel. Even compared with Transformer LM, ELECTRA is faster because it has a sigmoid layer for outputs. Transformer LM has a softmax layer, the computation of which is proportional to the vocabulary size.

3.2. Fine-tuning on ASR hypotheses

In pre-training, ELECTRA is trained to detect samples generated from BERT. However, in rescoring, ELECTRA needs to detect errors generated from the ASR model. BERT generates tokens on the basis of linguistic context, but the ASR model generates tokens on the basis of both acoustic features and linguistic context. Acoustically similar tokens, such as “two” and “too”, tend to be mistaken in ASR. To solve the mismatch between pre-training and rescoring conditions, we fine-tune ELECTRA on ASR hypotheses including real ASR errors. The ASR hypotheses are aligned with their corresponding references and each word of them is labeled as correct or incorrect. They can be obtained from paired speech and transcripts used for ASR training. Note that only the discriminator is fine-tuned.

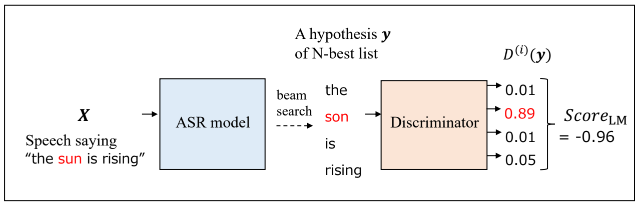


(a) Pre-training of ELECTRA



$\boldsymbol{p}$: HH IY HH AH Z EY P EH N
$\boldsymbol{p}^{\text{masked}}$: HH IY [MASK] AH Z EY [MASK] EH N
Encoder
Decoder
Generator (= Phone-to-Word)
$\boldsymbol{y}$
$\boldsymbol{y}^{\text{masked}}$
he
has
a
pen
he
[MASK]
a
pen
sample
$\boldsymbol{y}^{\text{corrupt}}$
he
was
a
pen
Discriminator (P-ELECTRA)
$D^{(i)}(\boldsymbol{y}^{\text{corrupt}})$
0 (original)
1 (replaced)
0 (original)
0 (original)

(b) Pre-training of Phone-attentive ELECTRA (P-ELECTRA)



$\boldsymbol{X}$
Speech saying “the sun is rising”
ASR model
beam search
A hypothesis $\boldsymbol{y}$ of N-best list
the
son
is
rising
Discriminator
$D^{(i)}(\boldsymbol{y})$
0.01
0.89
0.01
0.05
$Score_{\mathrm{LM}}$ = -0.96

(c) Rescoring with ELECTRA or P-ELECTRA

Fig. 1. Overview of our rescoring method. First, ELECTRA or P-ELECTRA is pre-trained on text data (a) or (b). It can then be fine-tuned on ASR hypotheses for error detection. Finally, it assigns $Score_{\mathrm{LM}}$ to each hypothesis for rescoring (c). ASR model can be regarded as generator that produces corrupted input.

3.3. Phone-attentive ELECTRA

Fine-tuning on ASR hypotheses can be done only on a limited amount of paired data, and therefore it is prone to overfitting. To generate acoustically similar errors on text data, we leverage phone information that can be automatically obtained from word sequences using a pronunciation dictionary [38, 39]. We propose a modified version of ELECTRA called phone-attentive ELECTRA (P-ELECTRA). In P-ELECTRA, the generator G is a phone-to-word conversion model instead of BERT used in original ELECTRA. It is a Transformer-based conditional masked language model (CMLM) [40], which consists of an encoder and a decoder, as illustrated in Fig. 1 (b). Some tokens of phones are randomly replaced with `[MASK]` and fed into the encoder as $\boldsymbol{p}^{\text{masked}}$ to prevent overfitting,

Table 1. Examples of generated tokens in pre-training of ELECTRA and P-ELECTRA. In addition to $\boldsymbol{y}^{\text{masked}}$, phone tokens are used as inputs in pre-training of P-ELECTRA. "_" denotes word boundary.

$\boldsymbol{y}$:	_i _don ' t _believe **_ann _knew** _any _magic _or _she ' d _have **_worked** _it _before
$\boldsymbol{y}^{\text{masked}}$:	_i _don ' t _believe `[MASK]` `[MASK]` _any _magic _or _she ' d _have `[MASK]` _it _before
$\boldsymbol{y}^{\text{corrupt}}$ in ELECTRA:	_i _don ' t _believe **_there _is** _any _magic _or _she ' d _have **_used** _it _before
$\boldsymbol{y}^{\text{corrupt}}$ in P-ELECTRA:	_i _don ' t _believe **_and _you** _any _magic _or _she ' d _have **_worked** _it _before

which is called "textaugment" [41]. Some tokens of words are also masked out and fed into the decoder as $\boldsymbol{y}^{\text{masked}}$. The generator predicts word tokens of the masked positions on the basis of both phone and word tokens and generates a corrupted example $\boldsymbol{y}^{\text{corrupt}}$.

$$\begin{aligned}\boldsymbol{y}^{\text{corrupt}} &= \text{replace}(\boldsymbol{y}^{\text{masked}}, \boldsymbol{m}, \hat{\boldsymbol{y}}) \\ \hat{y}_i &\sim p_G(y_i|\boldsymbol{y}^{\text{masked}}, \boldsymbol{p}^{\text{masked}}), i \in \boldsymbol{m}\end{aligned} \tag{9}$$

Note that phones are not required in rescoring because only the discriminator is used. For efficient training, sequences of phones and those of corresponding words are concatenated up to a certain length, respectively, then input to the generator.

Table 1 compares the generated tokens by BERT in the pre-training of ELECTRA and those by the phone-to-word conversion model in the pre-training of P-ELECTRA. In P-ELECTRA, "_ann _knew" is replaced with "_and _you" by using the phone-to-word conversion model, which is more acoustically similar and likely to appear in ASR results than the replacement by using BERT "_there _is".

3.4. Confidence estimation with ELECTRA

In ELECTRA (including P-ELECTRA), $c^{(i)}(\boldsymbol{y}) = 1 - D^{(i)}(\boldsymbol{y})$ can be regarded as a token-level confidence score. When we use subword units such as BPE, a word-level confidence score $c^{(i)}_{\text{word}}(\boldsymbol{y})$ is obtained by taking the minimum value of token-level scores for each word. ELECTRA leverages the knowledge of large text corpora for confidence estimation. BERT can also be applied to confidence estimation by pre-training for MLM then fine-tuning on ASR hypotheses. For fine-tuning, a new sigmoid layer for outputs is added to BERT, and its parameters are trained from scratch. On the other hand, ELECTRA has the sigmoid layer and can learn confidence scores even from the pre-training stage by detecting replaced tokens. ELECTRA is also more robust against ASR errors, as it is pre-trained not on ground truth text but on corrupted examples, which is indicated in punctuation restoration [42].

As mentioned in Section 2.3, the ASR model itself provides word-level confidence scores $p^{(i)}_{\text{word}}(\boldsymbol{y}|\boldsymbol{X})$. We interpolate the scores of ASR and ELECTRA as

$$c'^{(i)}_{\text{word}} = (1-\gamma)p^{(i)}_{\text{word}}(\boldsymbol{y}|\boldsymbol{X}) + \gamma c^{(i)}_{\text{word}}(\boldsymbol{y}) \tag{10}$$

where γ is a tunable weight parameter.

4. EXPERIMENTAL EVALUATIONS

4.1. Experimental conditions

We evaluated our method using the Librispeech [43] and TED-LIUM2 [44] corpora. The training data of Librispeech consists of 960 hours of paired speech and transcripts, and about 800 million words of additional text data. Its evaluation data consist of a "clean" set with lower-WER speakers and "other" set with higher-WER ones. The text is tokenized using BPE [36] of vocabulary size 9951 to make subword tokens. It is also converted to phone tokens of vocabulary size 74 using a lexicon. The training data of TED-LIUM2 consist of 207 hours of paired data and about 250 million words of additional text. The text is converted to subword tokens of vocabulary size 9798 and phone tokens of vocabulary size 44.

We prepared a CTC-based model for end-to-end ASR that consists of Transformer encoder with 12 layers, 256 hidden units, and 4 attention heads. We used the Adam [45] optimizer with Noam learning rate scheduling [3] of $warmup\_n = 25000, k = 5$. SpecAugment [46] was applied to acoustic features. Speed perturbation [47] was also applied in the TED-LIUM2 experiments. We obtained a 50-best list with beam search decoding.

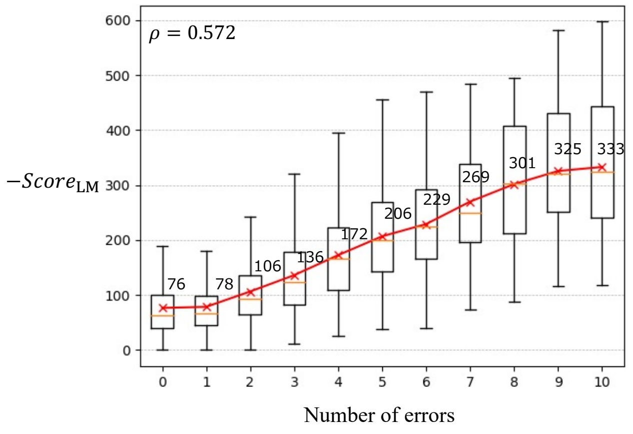
We prepared Transformer LM, BERT, ELECTRA, P-ELECTRA for rescoring and confidence estimation. Transformer LM, BERT, the generator for ELECTRA and ELECTRA (discriminator), and P-ELECTRA (discriminator) consist of Transformer with 12 layers, 256 hidden units, and 4 attention heads. The generator for P-ELECTRA (a phone-to-word conversion model) consists of 4-layer Transformer encoder and 4-layer Transformer decoder, which has almost the same number of parameters as those of other models. They are implemented with "transformers" library [48]. We pre-trained them on the Librispeech text data in Librispeech experiments and pre-trained them on the TED-LIUM2 text data in TED-LIUM2 experiments. Note that only discriminators are used for rescoring and confidence estimation tasks in ELECTRA and P-ELECTRA. We used the Adam optimizer with linear learning rate scheduling, in which the learning rate increases linearly for the first 10% of the total steps to 0.0001, thereafter decreasing linearly. During pre-training, 15% of the input subword tokens were selected and replaced with `[MASK]`. Next sentence prediction [13] is omitted from the pre-training objective of BERT in this study as in [49]. In the pre-training of P-ELECTRA, 30% of the phone tokens were also replaced with `[MASK]`, as mentioned in Section 3.3. We further fine-tuned BERT, ELECTRA, and P-ELECTRA for ASR error detection, or confidence estimation, using the 5-best list generated from the ASR training data. These fine-tuned models are denoted as BERT(FT), ELECTRA(FT), and P-ELECTRA(FT), respectively.

4.2. Experimental results

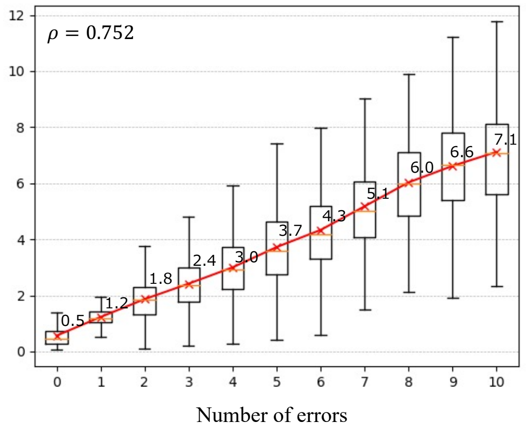
4.2.1. Rescoring on Librispeech

Table 2 compares rescoring results with different models on Librispeech. Transformer LM and BERT provide $Score_{\text{LM}}$ based on likelihood, as in Eqs. (6) and (7), respectively, and ELECTRAs (ELECTRA and P-ELECTRA) provide $Score_{\text{LM}}$ based on error counting, as in Eq. (8). α and β in Eq. (5) were adjusted using development sets. As the range of log-likelihood and that of error counts differ, α for Transformer LM or BERT and that for ELECTRAs were also different. For example, $\alpha = 0.5, \beta = 1.0$ were

Fig. 2. Scoring comparison with Transformer LM and fine-tuned P-ELECTRA for Librispeech "clean" set. "×" denotes average $-Score_{\text{LM}}$ for each "Number of errors".



(a) Scoring with Transformer LM



(b) Scoring with P-ELECTRA (FT)

Table 2. Rescoring results on Librispeech. x(FT) denotes that model x was fine-tuned on ASR 5-best list. "Runtime" denotes runtime compared with Transformer LM. Bottom 4 rows are results of proposed method.

	WER(%) ↓		Runtime ↓
	clean	other	clean
baseline ASR (CTC)	5.80	13.46	-
+Transformer LM	4.08	10.45	×1.0
+BERT	4.02	**10.25**	×27.1
+BERT (FT)	4.29	10.71	×0.8
+ELECTRA	4.07	10.87	×0.8
+ELECTRA (FT)	**3.98**	10.53	×0.8
+P-ELECTRA	4.04	10.44	×0.8
+P-ELECTRA (FT)	4.00	10.42	×0.8
oracle	2.47	7.85	-

Table 3. Correlation coefficients between LM scoring and number of errors. Scoring of Transformer LM and BERT is derived from likelihood (pseudo-likelihood), and scoring of bottom 5 methods is derived from error counting.

	Coefficients ρ	
	clean	other
Transformer LM	0.572	0.625
BERT	0.655	0.728
BERT (FT)	0.716	0.738
ELECTRA	0.620	0.634
ELECTRA (FT)	0.740	0.762
P-ELECTRA	0.687	0.767
P-ELECTRA (FT)	**0.752**	**0.788**

suitable for Transformer LM and $\alpha = 9.0, \beta = 0$ for ELECTRA. In terms of word error rate (WER), BERT outperformed Transformer LM, which is consistent with previous studies [14, 15]. ELECTRA pre-trained for replaced token detection detected ASR errors without fine-tuning and improved WER over the baseline ASR. By fine-tuning ELECTRA on ASR hypotheses (5-best), it outperformed Transformer LM for the "clean" set. Rescoring with P-ELECTRA outperformed ELECTRA and was competitive with Transformer LM without fine-tuning not only for the "clean" set but also for the "other" set. We saw that the pre-training method using phones in P-ELECTRA was effective, especially for the "other" set that contains more ASR errors. However, fine-tuning P-ELECTRA did not show much WER improvement. $Score_{\text{LM}}$ based on error counting in Eq. (8) can also be calculated with fine-tuned BERT, but this was not as successful as with ELECTRAs. We saw that the replaced token detection task for the pre-training of ELECTRAs is closer to the target rescoring task than MLM for BERT and thus lead to better rescoring. Our rescoring with ELECTRAs requires only 80% of the runtime of that with Transformer LM, as ELECTRAs do not compute the softmax layer of V entries. Rescoring with BERT resulted in the largest WER reduction especially for the "other" set, but its runtime was much inferior to the others, i.e. about 27 times slower than Transformer LM. It is dependent on not only V but also the length of a hypothesis L for each hypothesis.

Next, we compared $Score_{\text{LM}}$ of Transformer LM and that of fine-tuned P-ELECTRA for the Librispeech "clean" set, as shown in Fig. 2. "Number of errors" in the figure indicates *word-level* substitution, insertion, and deletion errors of each hypothesis against its reference, which are used in WER calculation. $-Score_{\text{LM}}$ of P-ELECTRA corresponds to the expected number of *subword-level* substitution and insertion errors. It is correlated to the number of all types of *word-level* errors. In P-ELECTRA, the average $-Score_{\text{LM}}$ increases almost linearly with the number of errors. Table 3 lists the correlation coefficients between $-Score_{\text{LM}}$ and the actual number of errors. Fine-tuning on actual ASR errors made the correlation higher. P-ELECTRA achieved a high correlation with and even without fine-tuning, which indicates its pre-training using phone information simulated ASR error detection well.

Table 4. Confidence estimation results on Librispeech. x(FT) denotes that model (CEM) x was fine-tuned on ASR 5-best list. ASR+x denotes confidence score interpolation between ASR and CEM x.

	AUC ↑		NCE ↑	
	clean	other	clean	other
ASR	0.955	0.927	0.546	0.426
BERT(FT)	0.945	0.914	0.544	0.394
ELECTRA	0.903	0.871	0.418	0.318
ELECTRA(FT)	0.956	0.928	0.594	0.455
P-ELECTRA	0.922	0.907	0.472	0.375
P-ELECTRA(FT)	**0.958**	**0.935**	**0.615**	**0.499**
ASR+BERT(FT)	0.973	0.952	0.660	0.535
ASR+ELECTRA	0.965	0.942	0.623	0.499
ASR+ELECTRA(FT)	0.976	0.956	0.681	0.560
ASR+P-ELECTRA	0.969	0.951	0.649	0.541
ASR+P-ELECTRA(FT)	**0.977**	**0.959**	**0.690**	**0.580**

4.2.2. *Confidence estimation on Librispeech*

We evaluated the performance of confidence estimation with different models (CEMs) on Librispeech, on the basis of the widely used metrics: the area under curve (AUC) of the receiver operating characteristic (ROC) curve and normalized cross entropy (NCE). The ROC curve shows the false positive and true positive rates for different thresholds. AUC values range from 0 to 1, and a higher AUC means a better estimator. Let $\boldsymbol{c} = (c_1, ..., c_N)$ denote estimated confidence scores for all words, and $\boldsymbol{t} = (t_1, ..., t_N)$ denote their corresponding targets, where $t_i = 1$ for correct words and 0 for incorrect words. NCE is defined as

$$NCE(\boldsymbol{c}, \boldsymbol{t}) = \frac{H(\boldsymbol{t}) - H(\boldsymbol{t}, \boldsymbol{c})}{H(\boldsymbol{t})} \tag{11}$$

where $H(\boldsymbol{t})$ denotes the entropy for the targets and $H(\boldsymbol{t}, \boldsymbol{c})$ denotes the binary cross entropy between the targets and estimated scores [50]. NCE measures how close the confidence scores are to the targets and NCE $= 1$ for the perfect estimator.

The results are listed in Table 4. Note that the same models are used in rescoring and confidence estimation. Confidence scores for "ASR" were obtained with the CTC forward-backward algorithm without any CEMs. ELECTRA and P-ELECTRA provided good confidence scores even without fine-tuning. By fine-tuning them, they outperformed fine-tuned BERT. BERT was pre-trained to predict masked tokens, while ELECTRAs were already pre-trained to detect inappropriate tokens. As shown in the bottom five rows of Table 4, we investigated interpolating confidence scores of ASR and CEMs, as in Eq. (10). γ was determined using the development sets. For example, $\gamma = 0.6$ was suitable for fine-tuned P-ELECTRA. By interpolation, we obtained far better confidence scores, which indicates CEMs pre-trained on large text corpora provide effective information that the ASR model does not provide. Among them, P-ELECTRA achieved the best performance.

4.2.3. *Rescoring and confidence estimation on TED-LIUM2*

We also conducted rescoring and confidence estimation experiments on TED-LIUM2. The results are listed in Table 5. They were evaluated on the "test" set, and the hyperparameters were adjusted using the "dev" set. In rescoring, similar trends to Librispeech were observed, and fine-tuned P-ELECTRA reduced WER as much as BERT with faster inference. In confidence estimation, CEMs themselves did not perform well, but the interpolation with ASR gave a large improvement. Among the CEMs, fine-tuned P-ELECTRA performed the best. On TED-LIUM2, the effect of fine-tuning was limited because of the smaller amount of paired data for generating the ASR 5-best list compared with Librispeech. Therefore, pre-training using phone information on text data was important for better performance.

Table 5. Rescoring and confidence estimation results on TED-LIUM2. "AUC(+ASR)" denotes AUC score by confidence score interpolation between ASR and CEM.

	Rescoring	Confidence estimation	
	WER ↓	AUC ↑	AUC(+ASR) ↑
ASR (CTC)	12.48	**0.914**	0.914
+Transformer LM	9.90	0.766	0.914
+BERT	**9.83**	0.854	0.914
+BERT (FT)	10.47	0.878	0.939
+ELECTRA	10.03	0.864	0.930
+ELECTRA (FT)	10.00	0.894	0.939
+P-ELECTRA	9.89	0.889	0.937
+P-ELECTRA (FT)	**9.83**	0.906	**0.942**
oracle	7.52		

5. CONCLUSIONS

We propose to apply ELECTRA to ASR rescoring and confidence estimation, in which ELECTRA detects ASR errors. ELECTRA is pre-trained on large text corpora to predict whether each token is replaced by BERT or not. However, there is a mismatch between ASR errors and token replacement by BERT. Fine-tuning on ASR hypotheses can eliminate this mismatch, and we further propose phone-attentive ELECTRA to mitigate the mismatch also in pre-training on text. In rescoring, we showed that ELECTRA was faster than Transformer LM because ELECTRA conducts binary classification with the sigmoid layer instead of the softmax computation, achieving competitive WER improvement. In confidence estimation, we also showed that fine-tuned ELECTRA worked better than fine-tuned BERT, and the interpolation with the ASR confidence provided highly reliable confidence scores. For future work, we will investigate corrupting inputs by not only replacement but also insertion and deletion in pre-training [51, 52] and predicting deletion errors [20] in rescoring.

6. REFERENCES

[1] Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber, "Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks," in *ICML*, 2006, pp. 369–376.

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

[3] Linhao Dong, Shuang Xu, and Bo Xu, "Speech-transformer: A

no-recurrence sequence-to-sequence model for speech recognition," in *ICASSP*, 2018, pp. 5884–5888.

[4] 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 *INTERSPEECH*, 2020, pp. 5036–5040.

[5] Alex Graves, "Sequence transduction with recurrent neural networks," *arXiv*, 2012.

[6] Qian Zhang, Han Lu, Hasim Sak, Anshuman Tripathi, Erik McDermott, Stephen Koo, and Shankar Kumar, "Transformer transducer: A streamable speech recognition model with transformer encoders and rnn-t loss," in *ICASSP*, 2020, pp. 7829–7833.

[7] Jan Chorowski and Navdeep Jaitly, "Towards better decoding and language model integration in sequence to sequence models," in *INTERSPEECH*, 2017, pp. 523–527.

[8] Kazuki Irie, Albert Zeyer, Ralf Schlüter, and Hermann Ney, "Language modeling with deep transformers," in *INTERSPEECH*, 2019, pp. 3905–3909.

[9] Anjuli Kannan, Yonghui Wu, Patrick Nguyen, Tara N. Sainath, ZhiJeng Chen, and Rohit Prabhavalkar, "An analysis of incorporating an external language model into a sequence-to-sequence model," in *ICASSP*, 2018, pp. 1–5828.

[10] Yuuki Tachioka and Shinji Watanabe, "Discriminative method for recurrent neural network language models," in *ICASSP*, 2015, pp. 5386–5390.

[11] Takaaki Hori, Chiori Hori, Shinji Watanabe, and John R. Hershey, "Minimum word error training of long short-term memory recurrent neural network language models for speech recognition," in *ICASSP*, 2016, pp. 5990–5994.

[12] Jiaji Huang, Yi Li, Wei Ping, and Liang Huang, "Large margin neural language model," in *EMNLP*, 2018, pp. 1183–1191.

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

[14] Joonbo Shin, Yoonhyung Lee, and Kyomin Jung, "Effective sentence scoring method using BERT for speech recognition," in *ACML*, 2019, pp. 1081–1093.

[15] Julian Salazar, Davis Liang, Toan Q. Nguyen, and Katrin Kirchhoff, "Masked language model scoring," in *ACL*, 2020, pp. 2699–2712.

[16] Wei Li, James Qin, Chung-Cheng Chiu, Ruoming Pang, and Yanzhang He, "Parallel rescoring with transformer for streaming on-device speech recognition," in *INTERSPEECH*, 2020, pp. 2122–2126.

[17] Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning, "ELECTRA: Pre-training text encoders as discriminators rather than generators," in *ICLR*, 2020.

[18] F. Wessel, R. Schluter, K. Macherey, and H. Ney, "Confidence measures for large vocabulary continuous speech recognition," *IEEE Transactions on Speech and Audio Processing*, pp. 288–298, 2001.

[19] Tatsuya Kawahara, Chin-Hui Lee, and B. Juang, "Flexible speech understanding based on combined key-phrase detection and verification," *IEEE Transactions on Speech and Audio Processing*, pp. 558–568, 1998.

[20] Anton Ragni, Qiujia Li, Mark Gales, and Yu Wang, "Confidence estimation and deletion prediction using bidirectional recurrent neural networks," in *SLT*, 2018, pp. 204–211.

[21] Prakhar Swarup, Roland Maas, Sri Garimella, Sri Harish Mallidi, and Björn Hoffmeister, "Improving ASR confidence scores for alexa using acoustic and hypothesis embeddings," in *INTERSPEECH*, 2019, pp. 2175–2179.

[22] Qiujia Li, David Qiu, Yu Zhang, Bo Li, Yanzhang He, Philip C. Woodland, Liangliang Cao, and Trevor Strohman, "Confidence estimation for attention-based sequence-to-sequence models for speech recognition," in *ICASSP*, 2021, pp. 6388–6392.

[23] Alejandro Woodward, Clara Bonnín, Issey Masuda, David Varas, Elisenda Bou-Balust, and Juan Carlos Riveiro, "Confidence measures in encoder-decoder models for speech recognition," in *INTERSPEECH*, 2020, pp. 611–615.

[24] Atsunori Ogawa, Naohiro Tawara, Takatomo Kano, and Marc Delcroix, "Blstm-based confidence estimation for end-to-end speech recognition," in *ICASSP*, 2021, pp. 6383–6387.

[25] Dan Oneaţă, Alexandru Caranica, Adriana Stan, and Horia Cucu, "An evaluation of word-level confidence estimation for end-to-end automatic speech recognition," in *SLT*, 2021, pp. 258–265.

[26] Alex Wang and Kyunghyun Cho, "BERT has a mouth, and it must speak: BERT as a Markov random field language model," in *Proceedings of the Workshop on Methods for Optimizing and Evaluating Neural Language Generation*, 2019, pp. 30–36.

[27] Ebru Arisoy, Abhinav Sethy, Bhuvana Ramabhadran, and Stanley Chen, "Bidirectional recurrent neural network language models for automatic speech recognition," in *ICASSP*, 2015, pp. 5421–5425.

[28] X. Chen, A. Ragni, X. Liu, and Mark J.F. Gales, "Investigating bidirectional recurrent neural network language models for speech recognition," in *INTERSPEECH*, 2017, pp. 269–273.

[29] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer, "Deep contextualized word representations," in *NAACL*, 2018, pp. 2227–2237.

[30] Kevin Clark, Minh-Thang Luong, Quoc Le, and Christopher D. Manning, "Pre-training transformers as energy-based cloze models," in *EMNLP*, 2020, pp. 285–294.

[31] Ankur Gandhe and Ariya Rastrow, "Audio-attention discriminative language model for asr rescoring," in *ICASSP*, 2020, pp. 7944–7948.

[32] Jilin Wang, Jiaji Huang, and Kenneth Ward Church, "Large margin training improves language models for ASR," in *ICASSP*, 2021, pp. 7368–7372.

[33] Daniel S. Park, Yu Zhang, Ye Jia, Wei Han, Chung-Cheng Chiu, Bo Li, Yonghui Wu, and Quoc V. Le, "Improved noisy student training for automatic speech recognition," in *INTERSPEECH*, 2020, pp. 2817–2821.

[34] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger, "On calibration of modern neural networks," in *ICML*, 2017, pp. 1321–1330.

[35] Shinji Watanabe, Takaaki Hori, Suyoun Kim, John R. Hershey, and Tomoki Hayashi, "Hybrid ctc/attention architecture for end-to-end speech recognition," *IEEE Journal of Selected Topics in Signal Processing*, pp. 1240–1253, 2017.

[36] Rico Sennrich, Barry Haddow, and Alexandra Birch, "Neural machine translation of rare words with subword units," in *ACL*, 2016, pp. 1715–1725.

[37] David Qiu, Qiujia Li, Yanzhang He, Yu Zhang, Bo Li, Liangliang Cao, Rohit Prabhavalkar, Deepti Bhatia, Wei Li, Ke Hu, Tara N. Sainath, and Ian McGraw, "Learning word-level confidence for subword end-to-end ASR," in *ICASSP*, 2021, pp. 6393–6397.

[38] Ryo Masumura, Naoki Makishima, Mana Ihori, Akihiko Takashima, Tomohiro Tanaka, and Shota Orihashi, "Phoneme-to-grapheme conversion based large-scale pre-training for end-to-end automatic speech recognition," in *INTERSPEECH*, 2020, pp. 2822–2826.

[39] Yun Tang, Juan Pino, Changhan Wang, Xutai Ma, and Dmitriy Genzel, "A general multi-task learning framework to leverage text data for speech to text tasks," in *ICASSP*, 2021, pp. 6209–6213.

[40] Marjan Ghazvininejad, Omer Levy, Yinhan Liu, and Luke Zettlemoyer, "Mask-predict: Parallel decoding of conditional masked language models," in *EMNLP*, 2019, pp. 6112–6121.

[41] Xiong Wang, Zhuoyuan Yao, Xian Shi, and Lei Xie, "Cascade rnn-transducer: Syllable based streaming on-device mandarin speech recognition with a syllable-to-character converter," in *SLT*, 2021, pp. 15–21.

[42] Michael Hentschel, Emiru Tsunoo, and Takao Okuda, "Making punctuation restoration robust and fast with multi-task learning and knowledge distillation," in *ICASSP*, 2021, pp. 7773–7777.

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

[44] Anthony Rousseau, Paul Deléglise, and Yannick Estève, "Enhancing the TED-LIUM corpus with selected data for language modeling and more TED talks," in *LREC*, 2014, pp. 3935–3939.

[45] Diederik P. Kingma and Jimmy Ba, "Adam: A method for stochastic optimization," *CoRR*, 2015.

[46] 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 *INTERSPEECH*, 2019, pp. 2613–2617.

[47] Tom Ko, Vijayaditya Peddinti, Daniel Povey, and S. Khudanpur, "Audio augmentation for speech recognition," in *INTERSPEECH*, 2015, pp. 3586–3589.

[48] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush, "Transformers: State-of-the-art natural language processing," in *EMNLP*, 2020, pp. 38–45.

[49] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, M. Lewis, Luke Zettlemoyer, and Veselin Stoyanov, "RoBERTa: A robustly optimized bert pretraining approach," *arXiv*, 2019.

[50] M. Siu and H. Gish, "Evaluation of word confidence for speech recognition systems," *Comput. Speech Lang.*, pp. 299–319, 1999.

[51] Jiatao Gu, Changhan Wang, and Junbo Zhao, "Levenshtein transformer," in *NeurIPS*, 2019.

[52] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer, "BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension," in *ACL*, 2020, pp. 7871–7880.