N-best T5: Robust ASR Error Correction using Multiple Input Hypotheses and Constrained Decoding Space

Rao Ma, Mark J. F. Gales, Kate M. Knill, Mengjie Qian

ALTA Institute, Machine Intelligence Lab, Department of Engineering, Cambridge University, UK

{rm2114,mjfg100,kmk1001,mq227}@cam.ac.uk

Abstract

Error correction models form an important part of Automatic Speech Recognition (ASR) post-processing to improve the readability and quality of transcriptions. Most prior works use the 1-best ASR hypothesis as input and therefore can only perform correction by leveraging the context within one sentence. In this work, we propose a novel N-best T5 model for this task, which is fine-tuned from a T5 model and utilizes ASR N-best lists as model input. By transferring knowledge from the pre-trained language model and obtaining richer information from the ASR decoding space, the proposed approach outperforms a strong Conformer-Transducer baseline. Another issue with standard error correction is that the generation process is not well-guided. To address this a constrained decoding process, either based on the N-best list or an ASR lattice, is used which allows additional information to be propagated.

Index Terms: speech recognition, error correction, ASR N-best list, path merging, lattice-constrained decoding

1. Introduction

Automated speech recognition (ASR) refers to the advanced technology of transcribing human speech into readable text. Early ASR systems employed HMM-based architectures where different modules such as the acoustic model, language model, and pronunciation lexicon, were trained separately and combined effectively during the decoding process. Recently, with the development of computing resources and availability of high-quality speech data, end-to-end (E2E) ASR models such as LAS and RNN-T have shown better performance and become more prevailing in both academia and industry [1, 2, 3]. Many successful applications have been built based on large-scale E2E ASR models with the aim of helping targeted users increase their productivity [4].

Although ASR systems have shown good performance in general, even beating human transcribers in terms of recognition accuracy in some experimental settings, they still face challenges when deployed in practice [5]. Therefore, error correction models that are expected to correct errors within the ASR output continue to serve an important role in post-processing [6]. Recently, end-to-end error correction models have shown promising performance on this task [7, 8, 9, 10]. Most works on error correction share a similar structure, taking the 1-best recognized hypothesis from the ASR system as input. The reference text is used as the target in training and beam search is used for generation in inference. We argue that

this standard practice does not provide enough information to the encoder and also gives the model too much freedom in decoding, which results in degraded performance.

The error correction model does not have access to the original utterance and makes predictions based on the ASR output. When the error correction model uses the 1-best hypothesis as input, it can only make predictions based on words within a single sentence. This sentence, however, contains limited information about the utterance and might be erroneous. Different strategies have been proposed to tackle this problem. Some works focus on obtaining and utilizing a more compact output from the ASR system such as the N-best list [11, 12], a word confusion net (WCN) [13], and a lattice [14, 15]. Other works propose using additional features such as phoneme sequences [16] in the correction process. These modifications enable the error correction model to obtain more useful information, making error detection and correction easier to perform.

To boost the performance of error correction models, we propose to fine-tune from a pre-trained language model (PLM) using concatenated N-best lists as input. With this extended input, the error correction model can effectively compare the difference between each hypothesis and becomes aware of the positions where the ASR system might have made mistakes. Since the oracle WER in the N-best list is relatively low, words within the N-best list also serve as a kind of "cues" to help the model recover the correct tokens. By fine-tuning from a high performing text-to-text transfer transformer (T5) model [17], we can take advantage of the structured knowledge implicitly learned from wide-ranging text data. As PLMs are pre-trained on large-scale text corpus via unsupervised learning, we hypothesize that these models can be easily adapted to plain text inputs, such as N-best lists, rather than more complex representations such as a lattice or WCN.

Error correction models usually use beam search in decoding, which we refer to as unconstrained decoding in this paper. For PLMs, words with similar meanings that share similar contexts are usually close to each other in the embedding parameter space. In some cases the model incorrectly generates synonyms that sound very different from the original utterance. To improve the robustness of the error correction model, we assume that it should only output hypotheses that also have a high ASR probability. To achieve this, we generate lattices in the decoding of the Transducer ASR model and propose several constrained decoding algorithms for the correction model.

The contributions of our work are as follows: This is the first paper in the error correction area to transfer knowledge from a pre-trained language model using N-best lists as input. Only a simple data augmentation method, SpecAugment, is needed for training data preparation. Compared to a strong Conformer-Transducer ASR baseline, we achieved a 7% WER

reduction on test sets by using N-best lists as input. To further improve model performance we introduce a novel lattice generation and conversion algorithm to the ASR Transducer model. By constraining the decoding space of the N-best T5 model we achieved up to 12% WER reduction on the test sets.

2. Discussion on ASR Output

The error correction model, as a downstream task of speech recognition, aims to correct errors contained in ASR transcriptions. Therefore, the input of the correction model is dependent on the output of the ASR system. In this section we discuss the different types of ASR outputs utilized in our experiments.

E2E ASR models typically employ the beam search algorithm to find the best decoding result. Suppose the beam size is n, at each time step a sentence list containing n-best partial hypotheses is maintained and updated. When the decoding process is completed for an utterance, we can get an ASR N-best list in addition to the 1-best recognition hypothesis.

A word lattice represents multiple ASR hypotheses and associated scores in a compact graph structure. It has been extensively exploited in HMM-based ASR systems [18, 19, 20]. Recent works attempt to produce lattices during the decoding of E2E ASR models [21, 22]. [21] introduces K-gram context approximation to RNN-T, which assumes that the prediction network only needs to rely on the previous $k - 1$ tokens instead of the entire predicted sentence history for effective estimation. Based on this assumption, a path-merging algorithm is proposed for beam search. At each decoding step, two partial hypotheses are merged if their last $k - 1$ tokens are identical. Then paths with lower scores are removed from the active beam but are kept in the lattice graph.
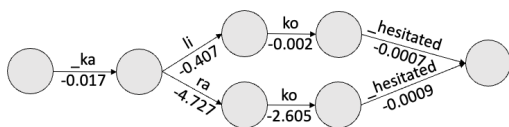

Figure 1: *An example of the generated ASR lattice.*

An example of a generated lattice with $k = 3$ is depicted in Figure 1. Past work has shown that the oracle WER in the lattice is reduced compared to the 1-best ASR hypothesis [21], however, whether the usage of the generated lattice can benefit downstream tasks remains to be demonstrated. In this paper, we show how the lattice produced in the Transducer decoding can help to improve the performance of the error correction system.

3. N-best T5 Model

3.1. Model Structure

A traditional error correction model uses the 1-best hypothesis generated by the ASR model as model input. With such limited information, it is difficult for the model to detect the errors or to make the right corrections. Meanwhile, an N-best list is always output by the ASR model as a byproduct of beam search. Hypotheses within this list are highly possible transcriptions of the original utterance and yield lower WER results compared to the 1-best hypothesis. By leveraging the diversified hypotheses contained in the N-best list, the error correction model can make more accurate and informed predictions.

The proposed N-best T5 error correction model is shown in Figure 2. It is based on a pre-trained T5 model with some modifications. N refers to how many top hypotheses we obtain from
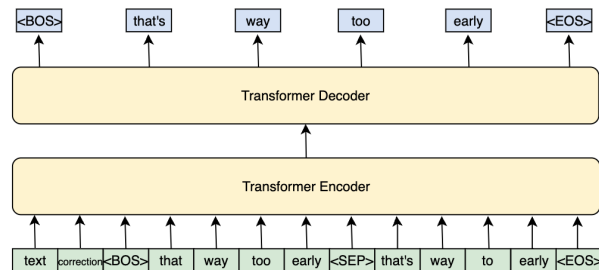

Figure 2: *Model structure of N-best T5.*

the ASR system, which is a hyper-parameter to be examined. In training and decoding, we concatenate the ASR N-best list and add a special token between different hypotheses to denote the sentence end. To comply with the pre-training practice of T5, a prefix "*text correction:*" is inserted to the input sequence. Preliminary experimental results show that without using the prefix, performance drops on the validation set.

Since the language model structure remains the same in the fine-tuning phase, the proposed method easily works with any PLM from third-party packages. Furthermore, as the N-best list can be easily obtained from ASR systems without the need of in-depth access, this error correction method can be applied to black-box ASR models integrated in cloud services.

3.2. Constrained Decoding in Inference

So far we have introduced how to incorporate more information into the encoder part of our proposed model. How to provide guidance for decoding and achieving controllable generation is another interesting question. The beam search algorithm is widely used as an approximation for finding the best generation result within the entire decoding space. This gives the model too much freedom, however, and we do not have much control over this decoding process [23]. For example, we hope the proposed model keeps correct words from the original transcription and only corrects incorrect words. In addition, the model is expected to output homophones of detected erroneous words. Although the model can implicitly learn these features from the training data, there is no guarantee in the decoding. For example, the model might output synonyms that have a high embedding similarity to words in the reference text. We propose several alternative decoding algorithms to better achieve the goal.

3.2.1. Unconstrained Decoding

The original decoding target is to find $\hat{y}$ that satisfies

$$\hat{y} = \arg\max_{y} \log P(y | \mathcal{Z}; \theta_{\text{EC}}) \quad (1)$$

where $\mathcal{Z} = \{\hat{z}^{(1)}, \hat{z}^{(2)}, \cdots, \hat{z}^{(n)}\}$ is the input to the N-best T5 encoder containing n ASR hypotheses. Since it is time-consuming to find the globally optimal sequence satisfying this equation, heuristic algorithms such as beam search are often used in decoding. Since no explicit constraints about the generated sequence are applied during decoding, we refer to this method as unconstrained decoding in this paper.

3.2.2. N-best Constrained Decoding

The original decoding space of the error correction model is unbounded. However, we expect the generated correction results to sound similar to the original utterance. The N-best list contains hypotheses generated by the ASR system which are most

likely to be the correct transcription. In N-best constrained decoding, we force the generation results to sentences within this ASR N-best list. Furthermore, since each path in the N-best list is also associated with a score calculated by the ASR system, we can combine the scores from two models with an interpolation weight λ. To be specific, $\hat{y}$ is the decoding result that maximizes Equation 2. $\boldsymbol{x}$ and $\boldsymbol{\mathcal{Z}}$ refer to the input acoustic features of the ASR system and the obtained ASR N-best list.

$$\hat{y} = \arg\max_{\boldsymbol{y} \in \boldsymbol{\mathcal{Z}}}[(1-\lambda) \cdot \log P(\boldsymbol{y}|\boldsymbol{x}; \boldsymbol{\theta}_{\text{ASR}}) \\ + \lambda \cdot \log P(\boldsymbol{y}|\boldsymbol{\mathcal{Z}}; \boldsymbol{\theta}_{\text{EC}})] \tag{2}$$

3.2.3. Lattice Constrained Decoding

An ASR N-best list can only encode a small subset of all the possible decoding results. Therefore, constraining the decoding result to appear in the N-best list might be a strict condition. Another data structure to consider is the lattice generated with path merging. Here, we restrict the decoding space to paths $\mathcal{G}$ in the lattice and combine ASR scores in the decoding.

$$\hat{y} = \arg\max_{\boldsymbol{y} \in \mathcal{G}}[(1-\lambda) \cdot \log P(\boldsymbol{y}|\boldsymbol{x}; \boldsymbol{\theta}_{\text{ASR}}) \\ + \lambda \cdot \log P(\boldsymbol{y}|\boldsymbol{\mathcal{Z}}; \boldsymbol{\theta}_{\text{EC}})] \tag{3}$$

Since the ASR model and the pre-trained language model use different tokenizers, we need to convert the original lattice into an equivalent form suitable for the N-best T5 to process. The lattice with ASR BPE tokens is first converted into a word lattice with dynamic programming as in Figure 3. The words on the edges are then split into BPE tokens by the tokenizer of T5, which is shown in Figure 4. The detailed decoding algorithm is shown in Algorithm 1, which is adapted from [24]. A similar algorithm is proposed in [14] while the lattice is obtained from different generation processes.
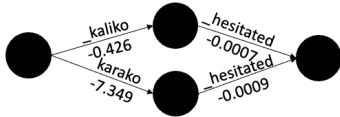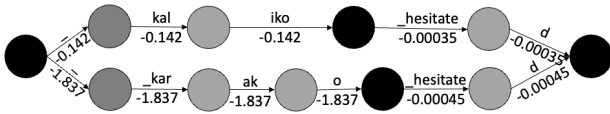

Figure 3: *Example lattice with words.*

Figure 4: *Example lattice with PLM BPE tokens.*

4. Experiments

4.1. Experiment Setup

We conduct experiments on the LibriSpeech dataset [25]. The data is collected from audiobook readings and the training set contains 960hr of speech data. We test on four subsets: dev_clean, dev_other, test_clean, and test_other, each containing ~5hr of speech. 80-dimensional log-mel filterbank features are extracted from the utterances and the sample rate is 16kHz.

The ASR model utilized in this paper adopts a novel Conformer-Transducer structure [3]. The encoder contains 12 Conformer layers with a hidden size of 512. The predictor has

---

Algorithm 1 Lattice Constrained Decoding for N-best T5

Data: lattice node set $\mathcal{V}$, lattice edge set $\mathcal{E}$, beam width b, T5 encoder outputs $\{\mathbf{h}_j\}$

1: $Q \leftarrow$ topological_sort($\mathcal{V}$)
2: **for** v in $\mathcal{V}$ **do**
3:     $H_v \leftarrow$ min_heap()
4: **end for**
5: n_0.history $= \epsilon, n_0$.score $= 0$
6: H_{start}.put(n_0)
7: **for** v in Q **do**
8:     $\mathbf{o} =$ Decoder($\{\mathbf{h}_j\}, n$.history, v.word)
9:     **for** $\langle v, x \rangle$ in $\mathcal{E}$ **do**
10:         **for** n in H_v **do**
11:             n'.history $=$ concat(n.history, v.word)
12:             n'.score $= n$.score$+\lambda \cdot \log(\mathbf{o}[x$.word$])+(1-\lambda) \cdot \log s_{vx}$
13:             **if** H_x.size $\geq b \wedge H_x$.score.min() $< n'$.score **then**
14:                 H_x.remove_min()
15:             **end if**
16:             **if** H_x.size $< b$ **then**
17:                 H_x.put(n')
18:             **end if**
19:         **end for**
20:     **end for**
21: **end for**
22: $I =$ max_heap(H_{end}.items)
23: **return** I.max()

---

one LSTM layer. The hidden dimensions of the jointer and predictor are both 512. SpecAugment [26] and speed perturbation are used for training data augmentation. The hyper-parameters of this Transducer model follow the ESPnet code base [27].

The error correction model learns from the erroneous transcriptions generated by the ASR model. Since the ASR model is trained on the training speech data, it shows an extremely low WER rate (close to zero) on this set. Data augmentation techniques are needed so that the error correction model generalizes to speech data unseen by the ASR model. Past works have used different data generation methods including decoding with under-performed ASR model [10], usage of dropout in the ASR decoding [7], and decoding on pseudo TTS data [28], etc.

In this work, we only use one simple yet effective data augmentation method, SpecAugment. Two frequency masks with mask parameter $F = 30$, eight time masks with mask parameter $T = 40$, and time warping with warp parameter $W = 40$ are utilized on the training speech data. We decode the ASR model on this perturbed data set and filter out sentences with WER higher than 0.25. The resulting training text corpus contains 262K sentence pairs. It is worth noting that in this work we do not leverage the extra text corpus of LibriSpeech. We assume that only the training speech is available for the error correction task, which is a more general case. Therefore, our training pipeline can be easily adapted to any speech corpus.

The error correction model is initialized using a pre-trained T5 base model. The model has 6 Transformer blocks for both the encoder and decoder with a hidden dimension of 768. We train the model on the perturbed training data for 3 epochs. AdamW [29] is utilized as the optimizer and the initial learning rate is 5e-5. In fine-tuning, the dropout rate is set to 0.1. A batch size of 32 is used for both training and testing. In the constrained decoding process, optimal interpolation weight λ is searched in the range of [0.0, 1.0] with a grid size of 0.05. In lattice constrained decoding, the beam width b is set to 1, and larger values yield similar results.

Table 1: *ASR system performance.*

Data	Dev		Test	
	clean	other	clean	other
Beam Search	2.71	6.99	2.88	7.06
+ Path Merging	2.70	7.01	2.91	7.13

4.2. Error Correction Results

In Table 1 we show the baseline results of the ASR model. Here we list the WER results of the 1-best decoding hypothesis obtained with and without path merging in the beam search. For both methods, a beam width of 10 is used in the decoding. Therefore, an N-best list containing the 10 best hypotheses is generated during decoding. For path merging, similar paths are merged with a context size of $k = 4$. The results are similar to [21] in that the performance of the ASR system decoded with the path-merging scheme shows comparable performance to the counterpart using full-context history.

Table 2: *Oracle WER results in ASR outputs.*

ASR Output	Dev		Test	
	clean	other	clean	other
5-best List	1.35	4.72	1.44	4.66
10-best List	1.24	4.43	1.34	4.34
Lattice	0.79	2.98	0.89	3.00

Table 2 shows the oracle WER results in different ASR outputs. The 5-best list contains the top 5 hypotheses from the 10-best list. The oracle WER in the 5-best list and 10-best list improve by 38.6% and 42.8% compared to the ASR baseline. Therefore, words within the N-best list can help the model to recover the correct transcription. By merging partial hypotheses with similar sentence history, we can generate word lattices in the decoding. As shown in the table, the lattice yields lower oracle WER results on the test sets. Results indicate that N-best lists and lattices can provide guidance in the decoder of the error correction model to achieve controllable decoding.

Table 3: *Results for ASR baseline and N-best T5 models with different model inputs.*

Model	Dev		Test	
	clean	other	clean	other
Baseline	2.71	6.99	2.88	7.06
1-best T5	2.89	6.94	3.02	7.18
5-best T5	2.62	6.40	2.77	6.67
10-best T5	**2.60**	**6.25**	**2.67**	**6.56**

The results of training an error correction model with different model inputs are listed in Table 3. In the decoding of the error correction model, the beam search algorithm with a beam size of 10 is used. From the table, we can see that the model fails to yield good results only using the 1-best decoding hypothesis as model input. Possible reasons have been discussed in this paper that without enough input information, the model fails to improve performance over a strong ASR system. For the 5-best T5 and the 10-best T5 model, 4.7% and 7.2% performance gains could be seen compared to the baseline ASR model. The results indicate that with a larger N, the error correction model obtains richer information from diversified input, which leads to more accurate error detection and correction.

4.3. Comparison of Decoding Algorithms

In this experiment, we compare the WER results of the N-best T5 model using different decoding algorithms. As shown in Table 4, by adding more constraints in the decoding process, the model performance is improved for both 5-best T5 and 10-best T5 models. With the proposed methods, we can guide the error correction model to generate homophones of the mistaken words. Since a lattice contains more possible paths compared to an N-best list and yields lower oracle WER, lattice-constrained decoding gives slightly better performance than N-best constrained decoding on the test sets.

Table 4: *Comparison of ASR baseline and N-best T5 model with different decoding algorithms.*

Model	Decoding Method	Dev		Test	
		clean	other	clean	other
Baseline	-	2.71	6.99	2.88	7.06
5-best T5	Unconstrained	2.62	6.40	2.77	6.67
	N-best Constrained	**2.38**	6.25	2.55	6.38
	Lattice Constrained	2.40	6.21	2.54	6.34
10-best T5	Unconstrained	2.60	6.25	2.67	6.56
	N-best Constrained	2.39	6.17	2.54	6.31
	Lattice Constrained	2.41	**6.11**	**2.53**	**6.27**

The effect of interpolation weight λ on the constrained decoding is depicted in Figure 5. λ is the weight of the error correction model. As the figure shows, constrained decoding with $\lambda = 1$ outperforms the plain beam search. In these cases, we constrain the error correction model to only generate hypotheses that are in the ASR N-best list or lattice rather than the entire decoding space. When $0 < \lambda < 1$, the scores calculated by the ASR system are linearly combined to bias the result towards paths with higher acoustic probabilities. The best performance is achieved when λ is around 0.75.
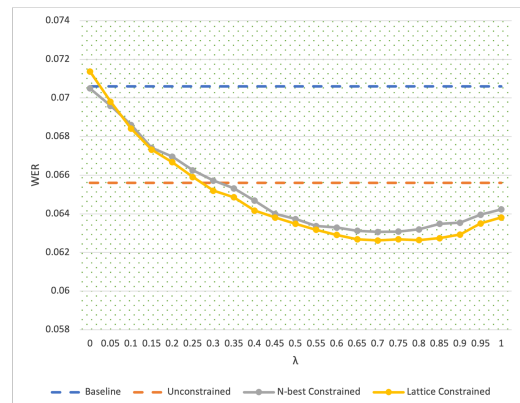
Figure 5: *Performance of 10-best T5 error correction model on test_other with different interpolation weights λ.*

5. Conclusions

In this work, we propose an effective error correction model based on transfer learning from pre-trained language models. The performance of the proposed method consistently improves over the performance of a strong ASR model on the LibriSpeech test sets. The two contributions of this paper: the usage of the N-best list as input to PLMs and constrained decoding algorithms based on the output from E2E ASR models can be applied to other models within the error correction area.

6. References

[1] A. Graves and N. Jaitly, "Towards end-to-end speech recognition with recurrent neural networks," in *Proc. International Conference on Machine Learning*. PMLR, 2014, pp. 1764–1772.

[2] W. Chan, N. Jaitly, Q. Le, and O. Vinyals, "Listen, attend and spell: A neural network for large vocabulary conversational speech recognition," in *Proc. 2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2016, pp. 4960–4964.

[3] A. Gulati, J. Qin, C.-C. Chiu, N. Parmar, Y. Zhang, J. Yu, W. Han, S. Wang, Z. Zhang, Y. Wu *et al.*, "Conformer: Convolution-augmented transformer for speech recognition," in *Proc. Interspeech 2020*, 2020, pp. 5036–5040.

[4] M. B. Hoy, "Alexa, Siri, Cortana, and more: an introduction to voice assistants," *Medical Reference Services Quarterly*, vol. 37, no. 1, pp. 81–88, 2018.

[5] A. Radford, J. W. Kim, T. Xu, G. Brockman, C. McLeavey, and I. Sutskever, "Robust speech recognition via large-scale weak supervision," *arXiv preprint arXiv:2212.04356*, 2022.

[6] R. Errattahi, A. El Hannani, and H. Ouahmane, "Automatic speech recognition errors detection and correction: A review," *Procedia Computer Science*, vol. 128, pp. 32–37, 2018.

[7] O. Hrinchuk, M. Popova, and B. Ginsburg, "Correction of automatic speech recognition with transformer sequence-to-sequence model," in *Proc. 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 7074–7078.

[8] S. Zhang, M. Lei, and Z. Yan, "Automatic spelling correction with transformer for CTC-based end-to-end speech recognition," *arXiv preprint arXiv:1904.10045*, 2019.

[9] S. Dutta, S. Jain, A. Maheshwari, G. Ramakrishnan, and P. Jyothi, "Error correction in ASR using sequence-to-sequence models," *arXiv preprint arXiv:2202.01157*, 2022.

[10] Y. Zhao, X. Yang, J. Wang, Y. Gao, C. Yan, and Y. Zhou, "BART based semantic correction for Mandarin automatic speech recognition system," in *Proc. Interspeech 2021*, 2021.

[11] L. Zhu, W. Liu, L. Liu, and E. Lin, "Improving ASR error correction using n-best hypotheses," in *Proc. 2021 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*. IEEE, 2021, pp. 83–89.

[12] Y. Leng, X. Tan, R. Wang, L. Zhu, J. Xu, W. Liu, L. Liu, X.-Y. Li, T. Qin, E. Lin *et al.*, "Fastcorrect 2: Fast error correction on multiple candidates for automatic speech recognition," in *Proc. Findings of the Association for Computational Linguistics: EMNLP 2021*, 2021, pp. 4328–4337.

[13] Y. Weng, S. S. Miryala, C. Khatri, R. Wang, H. Zheng, P. Molino, M. Namazifar, A. Papangelis, H. Williams, F. Bell *et al.*, "Joint contextual modeling for ASR correction and language understanding," in *Proc. 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 6349–6353.

[14] R. Ma, H. Li, Q. Liu, L. Chen, and K. Yu, "Neural lattice search for speech recognition," in *Proc. 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 7794–7798.

[15] L. Dai, L. Chen, Z. Zhou, and K. Yu, "LatticeBART: Lattice-to-lattice pre-training for speech recognition," in *Proc. 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2022, pp. 6112–6116.

[16] H. Wang, S. Dong, Y. Liu, J. Logan, A. K. Agrawal, and Y. Liu, "ASR error correction with augmented transformer for entity retrieval," in *Proc. Interspeech 2020*, 2020.

[17] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, "Exploring the limits of transfer learning with a unified text-to-text transformer," *Journal of Machine Learning Research*, vol. 21, no. 1, pp. 5485–5551, 2020.

[18] H. Xu, T. Chen, D. Gao, Y. Wang, K. Li, N. Goel, Y. Carmiel, D. Povey, and S. Khudanpur, "A pruned RNNLM lattice-rescoring algorithm for automatic speech recognition," in *Proc. 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 5929–5933.

[19] P. Zhang, B. Chen, N. Ge, and K. Fan, "Lattice transformer for speech translation," in *Proc. 57th Annual Meeting of the Association for Computational Linguistics*, 2019, pp. 6475–6484.

[20] X. Liu, X. Chen, Y. Wang, M. J. Gales, and P. C. Woodland, "Two efficient lattice rescoring methods using recurrent neural network language models," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 24, no. 8, pp. 1438–1449, 2016.

[21] R. Prabhavalkar, Y. He, D. Rybach, S. Campbell, A. Narayanan, T. Strohman, and T. N. Sainath, "Less is more: Improved RNN-T decoding using limited label context and path merging," in *Proc. 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 5659–5663.

[22] M. Novak, P. Papadopoulos, and A. A. AI, "RNN-T lattice enhancement by grafting of pruned paths," in *Proc. Interspeech 2022*, 2022, pp. 4960–4964.

[23] S. Dathathri, A. Madotto, J. Lan, J. Hung, E. Frank, P. Molino, J. Yosinski, and R. Liu, "Plug and play language models: A simple approach to controlled text generation," in *Proc. International Conference on Learning Representations*, 2020.

[24] M. Auli, M. Galley, C. Quirk, and G. Zweig, "Joint language and translation modeling with recurrent neural networks," in *Proc. EMNLP*, 2013.

[25] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: an ASR corpus based on public domain audio books," in *Proc. 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2015, pp. 5206–5210.

[26] D. S. Park, W. Chan, Y. Zhang, C.-C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le, "SpecAugment: A simple data augmentation method for automatic speech recognition," in *Proc. Interspeech 2019*, 2019, pp. 2613–2617.

[27] S. Watanabe, T. Hori, S. Karita, T. Hayashi, J. Nishitoba, Y. Unno, N.-E. Y. Soplin, J. Heymann, M. Wiesner, N. Chen *et al.*, "ESPnet: End-to-End Speech Processing Toolkit," in *Proc. Interspeech 2018*, 2018, pp. 2207–2211.

[28] J. Guo, T. N. Sainath, and R. J. Weiss, "A spelling correction model for end-to-end speech recognition," in *Proc. 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 5651–5655.

[29] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," in *Proc. International Conference on Learning Representations*, 2019.