End-to-end Joint Punctuated and Normalized ASR with a Limited Amount of Punctuated Training Data

Can Cui, Imran Ahamad Sheikh, Mostafa Sadeghi, Emmanuel Vincent

END-TO-END JOINT PUNCTUATED AND NORMALIZED ASR WITH A LIMITED AMOUNT OF PUNCTUATED TRAINING DATA

Can Cui[1,2]*, Imran Sheikh*[2]*, Mostafa Sadeghi*[1]*, Emmanuel Vincent*[1]

[1]Université de Lorraine, CNRS, Inria, LORIA, F-54000 Nancy, France
[2]Vivoka, Metz, France

ABSTRACT

Joint punctuated and normalized automatic speech recognition (ASR), that outputs transcripts with and without punctuation and casing, remains challenging due to the lack of paired speech and punctuated text data in most ASR corpora. We propose two approaches to train an end-to-end joint punctuated and normalized ASR system using limited punctuated data. The first approach uses a language model to convert normalized training transcripts into punctuated transcripts. This achieves a better performance on out-of-domain test data, with up to 17% relative Punctuation-Case-aware Word Error Rate (PC-WER) reduction. The second approach uses a single decoder conditioned on the type of output. This yields a 42% relative PC-WER reduction compared to Whisper-base and a 4% relative (normalized) WER reduction compared to the normalized output of a punctuated-only model. Additionally, our proposed model demonstrates the feasibility of a joint ASR system using as little as 5% punctuated training data with a moderate (2.42% absolute) PC-WER increase.

***Index Terms*—** ASR, punctuated transcripts, RNN-T, limited data, streaming ready

1. INTRODUCTION

Transcribing speech into text with punctuation and casing has been an active area of research in ASR [1–4]. According to some definitions,[1] casing is considered as part of punctuation. For notational simplicity, we therefore refer to punctuated-cased ASR/transcripts as *punctuated* ASR/transcripts in the rest of this paper. Joint punctuated and normalized ASR, which produces transcripts both with and without punctuation and casing, is highly desirable because (a) it improves human readability (b) it extends compatibility with natural language processing models that either exploit or discard punctuation information, and (c) it simplifies model deployment and maintenance.

The conventional approach to punctuated ASR involves post-processing the output of a normalized ASR system with a punctuation and case restoration model [5–8]. For instance, the authors in [8] used a modified Recurrent Neural Network-Transducer (RNN-T) ASR model [9] and a fine-tuned ELECTRA language model (LM) [10]. While this approach does not require punctuated transcripts for ASR training, it increases the total model size and inference time. Moreover, it does not exploit acoustic information for punctuation. The second conventional approach employs an end-to-end (E2E) ASR model to directly generate punctuated transcripts [11, 12]. A typical punctuated ASR system is Whisper [13], which is trained on large-scale punctuated data. E2E punctuated ASR models are typically less accurate at word recognition compared to equal-sized normalized ASR models, resulting in a poorer normalized ASR performance [14, 15] and increased computation time when output normalization is needed. To mitigate this, the study in [14] introduced an auxiliary Connectionist Temporal Classification (CTC) loss for transcribing normalized text at the output of an intermediate layer of the E2E model. The study in [15] proposed three decoders that output spoken, written, and joint transcripts. While these methods achieve joint punctuated and normalized ASR without increasing model size, they require a fully punctuated ASR corpus for training.

Beyond just punctuation and casing, the study in [16] introduced a transducer-based ASR system that also performs Inverse Text Normalization (ITN). This system requires a large inverse normalized training set. Moreover, it requires a longer context during inference [16, 17], resulting in a longer output latency and/or a larger real-time factor (RTF). By contrast, our aim is to design an E2E joint punctuated and normalized ASR system that is well-suited for low latency, low RTF applications.

Classically, punctuation and casing were not considered as part of the ASR task. Therefore, most ASR training corpora include normalized transcripts only and cannot be readily used to train punctuated ASR models. For instance, the 900 h CGN [18] corpus is one of the biggest corpora available for training Dutch ASR, yet it comes with normalized transcripts. The 100 h Dutch CommonVoice corpus [19] has punctuated transcripts, albeit from isolated sentences which are not suitable for training models that generalize well to long utterances. Audiobooks are a major source of punctu-

[1]https://dictionary.cambridge.org/grammar/british-grammar/punctuation

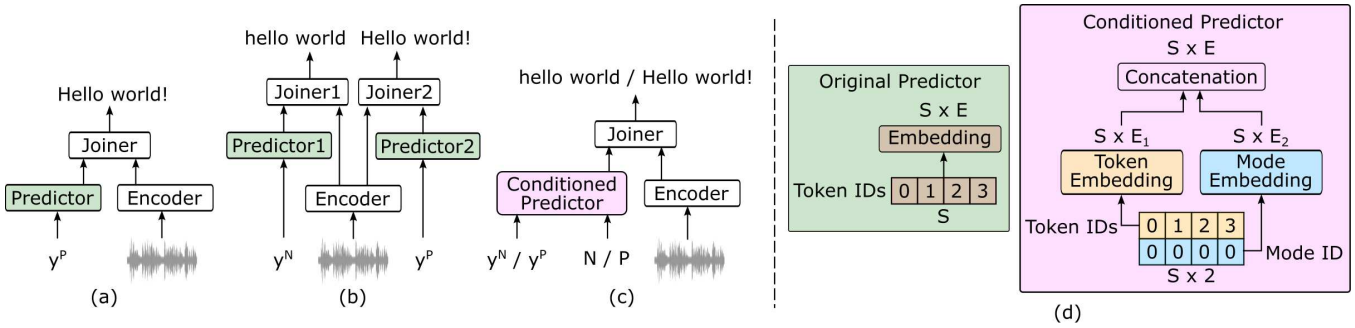

Fig. 1: (a) Stateless transducer-based punctuated ASR, (b) 2-Decoder (Joiner + Predictor) joint normalized-punctuated ASR, (c) the proposed conditioned Predictor ASR, (d) input layers in the original and conditioned Predictors.

ated long utterances, however producing usable ASR corpora out of this data is nontrivial. For instance, the Multilingual Librispeech corpus [20] includes 1,500 h of Dutch data but from 40 speakers only, and the transcripts have been normalized. In summary, the significant imbalance between punctuated and normalized ASR training data adds a challenge in training an E2E joint punctuated and normalized ASR model.

This paper aims for an E2E joint punctuated and normalized ASR system that is (a) efficient at punctuated as well as normalized transcription tasks, (b) trainable with a limited amount of punctuated labeled data, and (c) suitable for streaming applications. We introduce and compare two complementary approaches to train a stateless transducer-based E2E joint punctuated and normalized ASR model. The first approach uses an LM to generate punctuated training transcripts. However, such LMs may not be accurate enough or available for certain domains [21] and/or languages. To address such scenarios, we propose a second approach in which a single decoder is conditioned on the type of output. Experimental results show that our first method results in a 17% relative error reduction, while the second method enables training with an exceptionally low proportion of punctuated data.

The paper is organized as follows. Section 2 presents the preliminaries. Section 3 introduces our approaches. Section 4 describes our experiments. We conclude in Section 5.

2. PRELIMINARIES

2.1. Stateless transducer-based punctuated ASR

An E2E punctuated ASR system [14] directly transcribes speech into a punctuated transcript. Given an acoustic feature sequence $X \in \mathbb{R}^{L \times A}$ where L is the sequence length and A the feature dimension, the training objective is to maximize the probability

$$P(Y^{\mathrm{P}}|X) = \prod_{s=1}^{S^{\mathrm{P}}} P(y_s^{\mathrm{P}}|y_{[1:s-1]}^{\mathrm{P}}, X) \tag{1}$$

by generating a sequence $Y^{\mathrm{P}} \in \mathbb{R}^{S^{\mathrm{P}}}$, where "P" stands for punctuated transcription, and S^{P} represents the punctuated sequence length. The loss function can be written as

$$\mathcal{L}^{\mathrm{P}} = -\sum_{Y^{\mathrm{P}},X} \log P(Y^{\mathrm{P}}|X). \tag{2}$$

A stateless transducer-based E2E ASR system, which uses an RNN-T framework [22] with a stateless prediction network [9], can be readily extended to the punctuated transcription task (see Figure 1(a)). This brings the natural streaming recognition capability of RNN-T to punctuated ASR.

2.2. 2-Decoder joint normalized-punctuated ASR

Inspired by [14, 15], we design an ASR system with two Decoders, each consisting of its own Predictor and Joiner (see Figure 1(b)). Using the output of the same Encoder, these two Decoders generate the punctuated and normalized transcripts. Their respective training objectives are[2]

$$P(Y^{\mathrm{N}}|X) = \prod_{s=1}^{S^{\mathrm{N}}} P(y_s^{\mathrm{N}}|y_{[1:s-1]}^{\mathrm{N}}, X), \tag{3}$$

$$P(Y^{\mathrm{P}}|X) = \prod_{s=1}^{S^{\mathrm{P}}} P(y_s^{\mathrm{P}}|y_{[1:s-1]}^{\mathrm{P}}, X), \tag{4}$$

where "N" stands for normalized transcription, and S^{N} represents the normalized sequence length. The joint loss function is defined as

$$\mathcal{L}^{\text{2-decoder}} = \mathcal{L}^{\mathrm{N}} + \mathcal{L}^{\mathrm{P}} \tag{5}$$

$$= -\sum_{Y^{\mathrm{N}},X} \log P(Y^{\mathrm{N}}|X) - \sum_{Y^{\mathrm{P}},X} \log P(Y^{\mathrm{P}}|X). \tag{6}$$

[2]These equations are kept consistent with (1) and do not account for the stateless and/or streaming operation of the transducer model.

3. PROPOSED METHODS

3.1. Training ASR using auto-punctuated transcripts

Automatically generated punctuated transcripts of ASR training data can be used to train punctuated ASR models in the absence of human-generated punctuated transcripts. To the best of our knowledge, the use of auto-punctuated transcripts to train punctuated ASR models has been unexplored in prior works. The work in [12] proposed a semi-supervised rich ASR training method in which a rich ASR system is first trained on a small amount of human-labeled rich training data and then used to automatically generate auto-rich transcripts of a larger unlabeled speech corpus. This approach focused on the rich transcription of speech phenomena such as fillers, laughter, coughs, etc., hence some human-labeled data is necessary. By contrast, our transcription task focuses on punctuation and casing. Punctuation- and case-enhanced transcripts can be directly obtained from normalized transcripts using state-of-the-art punctuation and case restoration models [5–8]. Interestingly, this approach can be applied in scenarios where there is no punctuated labeled training data at all.

3.2. Conditioned Predictor ASR

A drawback of the auto-punctuated transcription-driven ASR training approach presented in Section 3.1 is that errors made by the punctuation and case prediction model may propagate into the punctuated ASR model. Moreover, such case and punctuation restoration models may not be accurate enough, or even available, for certain domains [21] and/or languages. Hence, we propose a second training approach to address such scenarios.

Our approach effectively utilizes the small amount of punctuated training data and the large amount of normalized training data by using a single conditioned Predictor to handle punctuated and normalized transcriptions. As shown in Figure 1(c), the Predictor is conditioned on the transcription mode ID, which is an input that specifies whether we want a normalized (N) or punctuated (P) output. The mode ID input is inspired by the language ID information used by E2E multilingual ASR models [23, 24]. As shown in Figure 1 (d), the token embeddings are concatenated with the mode embedding before feeding them to the following Predictor layers. The Conditioned Predictor ASR system can use a loss function similar to Equation (4). However, only part of the input samples will have a punctuated reference transcript and will use the corresponding loss. This could lead to uneven model performance between the two output modes, particularly poor performance of the punctuated transcription mode. To address this issue, we employ a tradeoff parameter α to adjust the weights of $\mathcal{L}^{\text{N}}$ and $\mathcal{L}^{\text{P}}$, as follows:

$$\mathcal{L}^{\text{cond-predictor}} = (1-\alpha)\mathcal{L}^{\text{N}} + \alpha\mathcal{L}^{\text{P}}. \tag{7}$$

4. EXPERIMENTS

4.1. Dataset and metrics

4.1.1. Dataset

We use the LibriSpeech corpus [25] to conduct our experiments. Following the standard split, our training set is the entire train-960 set. Similarly, the validation sets consist of dev-clean and dev-other, while the test sets are test-clean and test-other. We retrieve the punctuated transcripts of the LibriSpeech corpus from the original Project Gutenberg texts.[3] We have removed certain erroneous sentences, such as chapter names where all letters are in uppercase. A recent work on building a punctuated Librispeech corpus [26] has detailed retrieval steps similar to those we have developed independently. To evaluate the model's performance on real unseen data, we use the CommonVoice [27] English clean test set and independent headset microphone (IHM) recordings of the AMI meeting corpus [28] test set along with their original punctuated transcripts.

4.1.2. Metrics

The regular Word Error Rate (WER) metric used in ASR evaluation is insufficient to assess the performance in punctuated transcription tasks. Classically, Precision, Recall, and F1 score metrics have been used to evaluate the performance of punctuated transcription [7, 8]. However, these prior works present separate performance figures for each punctuation mark and lack a standardized metric for casing. Recently, some adapted metrics were proposed in [26] for calculating error rates with punctuation and case, highlighting the need for new metrics. However, error rates for case are lacking and punctuation error rates include word errors, which does not reflect errors solely caused by punctuation marks. To have a better understanding of the overall performance of the prediction of punctuation marks as well as casing, and to facilitate the comparison of error rates across words, punctuation, and casing, we introduce the concepts of Punctuation-only Error Rate (PuncER), Case-only Error Rate (CaseER) and Punctuation-Case-aware WER (PC-WER) alongside the WER. These metrics are calculated as:

$$\text{PuncER} = \frac{E_{\text{p-nc}} - E_{\text{np-nc}}}{N_{\text{p}}}, \quad \text{CaseER} = \frac{E_{\text{np-c}} - E_{\text{np-nc}}}{N_{\text{c}}},$$
$$\text{PC-WER} = \frac{E_{\text{p-c}}}{N_{\text{p-c}}}, \qquad \text{WER} = \frac{E_{\text{np-nc}}}{N_{\text{np-nc}}}, \tag{8}$$

where E represents the total number of substitution, deletion, and insertion errors, and N represents the total number of words, punctuation and/or casing marks. Suffixes "p" and "c" stand for the presence of punctuation and casing whereas suffices "np" and "nc" stand for the absence of punctuation

[3] available at `https://www.openslr.org/12/`

Table 1: Error rates (%) on in-domain† test data for all the ASR models trained on LibriSpeech train-960 with different types of transcripts (Trans.): normalized (N), original punctuated (P), or auto-punctuated (P'). All the test ground truth transcripts are the original punctuated transcripts. The 2-Decoder ASR and the Conditioned Predictor ASR systems use both punctuated (P or P') and normalized (N) transcripts for the entire train-960 set.

Type of ASR	# Param	Transcrip	LibriSpeech test-clean				LibriSpeech test-other				RTF
			WER	PuncER	CaseER	PC-WER	WER	PuncER	CaseER	PC-WER	
Whisper-base	74 M	-	5.80	33.86	32.04	15.96	13.02	36.79	27.72	19.48	**0.09**
Whisper-small	244 M	-	4.40	31.19	28.49	10.95	11.89	32.17	25.67	17.59	0.23
Cascaded (ASR+LM)	180 M	N	**2.18**	44.33	35.60	11.70	**5.03**	47.69	34.82	14.91	0.79
Punctuated only (a)	70 M	P	2.45	**30.39**	28.21	**9.12**	5.73	30.51	**25.14**	11.94	0.45
	70 M	P'	2.35	44.43	31.42	11.60	5.49	47.39	30.14	14.94	0.44
2-Decoder (b)	71 M	P + N	**2.33**	**29.99**	**26.99**	**8.85**	5.45	31.05	**24.76**	**11.76**	0.58
	71 M	P'+ N	**2.28**	45.10	31.68	11.67	**5.38**	47.81	29.53	14.88	0.59
Cond Predictor (c)	70 M	P + N	**2.35**	**29.14**	35.61	**9.32**	5.49	**29.62**	30.77	12.01	0.44
	70 M	P'+ N	**2.25**	44.60	41.86	12.30	**5.24**	46.84	33.99	15.42	0.45

†: For Whisper, it is unknown whether the test data is in-domain or out-of-domain due to the lack of training data information.
Note: We employed the SCTK toolkit [29] to conduct Matched Pair Sentence Segment statistical significance tests adapted to the four metrics. In all the tables, we highlight in bold the best result in each column and the results statistically equivalent to it at a 0.05 significance level.

Table 2: Example of evaluation metrics computed for reference **Hi, I am Chloe.** and hypothesis **hey I am chloe.**

WER	PuncER	CaseER	PC-WER
$1/4 = 0.25$	$(2-1)/2 = 0.5$	$(2-1)/2 = 0.5$	$3/6 = 0.5$

and casing, respectively. Table 2 shows an example of these metrics computed on a sample reference and prediction.

To measure the inference speed from a streaming perspective, we use the Real-Time Factor (RTF) as an additional metric, calculated as inference time divided by audio length. For Whisper models, we load the models once and process test audio files one by one. For our transducer-based models, we export the model to the ONNX format and similarly process the test audio files sequentially. The RTFs are computed during the inference process for obtaining the punctuated transcription on CPU.

4.2. Model and training setup

We use 80-dimensional log Mel filterbank features as input to all ASR models. Our tokenizer is a SentencePiece model [30] with a vocabulary size of 500. We use pretrained Whisper models [13] as baseline systems, employing Whisper-base (74M parameters) for a comparable number of parameters to our model, and Whisper-small (244M parameters) as a high-performance comparison. All our ASR models (E2E ASR, 2-Decoder ASR, Conditioned Predictor ASR) were trained using the recipe for the pruned stateless-transducer based on the Zipformer [31] encoder in the icefall toolkit.[4] For Conditioned Predictor ASR, the Token Embedding E_1 and Mode

[4] https://github.com/k2-fsa/icefall/blob/master/egs/librispeech/ASR/pruned_transducer_stateless7

Table 3: Examples of paired original punctuated transcripts and auto-punctuated transcripts from LibriSpeech.

Original	Auto-punctuated
See, I could even get in myself!	See I could even get in myself.
A man said to the universe: Sir, I exist!	A man said to the universe sir, I exist.
A red-haired mountain of a man, with an apparently inexhaustible store of energy.	A red haired mountain of a man with an apparently inexhaustible store of energy.

Embedding E_2 have 500 and 12 dimensions, respectively. All the models were trained for 40 epochs using 4 Nvidia RTX 2080 Ti GPUs, and inference was performed on an Intel Xeon Gold 5218R CPU.

We use the Rpunct [32] model to generate punctuated transcripts from normalized transcripts. This model is derived from the BERT [33] LM after fine-tuning it for English punctuation and case restoration tasks. Given a lower-cased text, the model can restore upper-casing, period (“.”), exclamation (“!”), question mark (“?”), comma (“,”), colon (“:”), semi-colon (“;”), apostrophe (“”) and dash (“-”) marks. The generated auto-punctuated transcripts may differ from the expected punctuated transcripts, as illustrated by some sample LibriSpeech sentences in Table 3.

4.3. Evaluation results

4.3.1. Effectiveness of E2E models

Table 1 presents the error rates on in-domain test sets of our ASR models. First of all, the transducer-based ASR models demonstrate superior performance compared to Whisper

Table 4: Error rates (%) on out-of-domain† test sets for all the ASR models.

Type of ASR	Transcrip	CommonVoice test-clean				AMI test			
		WER	**PuncER**	**CaseER**	**PC-WER**	**WER**	**PuncER**	**CaseER**	**PC-WER**
Whisper-base	-	38.06	34.03	11.09	38.99	71.72	64.22	39.84	73.27
Whisper-small	-	37.90	36.72	**10.56**	39.15	58.49	52.99	36.48	60.12
Cascaded (ASR+LM)	N	**27.90**	**27.91**	11.63	**29.50**	**37.46**	74.84	45.92	**45.99**
Punctuated only (a)	P	29.23	64.87	16.47	36.61	39.59	97.02	38.57	50.61
	P'	**27.81**	**27.28**	11.42	**29.27**	39.01	71.82	37.39	46.33
2-Decoder (b)	P + N	28.44	64.35	16.04	35.79	39.12	96.48	39.44	50.18
	P'+ N	28.35	**27.65**	**10.83**	**29.70**	39.33	**65.63**	**33.64**	**45.43**
Cond Predictor (c)	P + N	28.57	60.35	23.21	36.35	40.70	96.40	38.40	51.47
	P'+ N	**27.84**	**27.44**	11.34	**29.65**	38.77	65.65	35.62	**45.24**

†: For Whisper, it is unknown whether the test data is in-domain or out-of-domain due to the lack of training data information.

models. Specially, our proposed Conditioned Predictor ASR achieves a PC-WER reduction of up to 42% relative (from 15.96% to 9.32%) compared to Whisper-base, which is of comparable size, and 15% relative (from 10.95% to 9.32%) compared to Whisper-small. For the transducer-based ASR models, the conventional cascaded system, which uses a normalized output ASR model followed by a punctuation and case restoration LM, is included as a baseline. Comparison of error rates of the cascaded system and E2E models trained using punctuated data shows that the latter can lead to significant reductions in PC-WER. For instance, when using the 2-Decoder ASR model (P + N) we obtain a relative CP-WER reduction of up to 24% (from 11.70% to 8.85%).

When comparing the 2-Decoder ASR model with the proposed Conditioned Predictor ASR model on the LibriSpeech test set, the latter exhibits a decrease in PuncER. For example, when using the same P + N training set, the Conditioned Predictor ASR model achieves a relative decrease of 3% on test-clean and 5% on test-other in PuncER compared to the former. But 2-Decoder ASR has a better performance in terms of case generation, by reducing the CaseER by 24% relative on test-clean (from 35.60% to 26.99%).

Furthermore, the proposed Conditioned Predictor ASR gives both normalized and punctuated outputs. The normalized output shows a 4% relative reduction in WER on test-clean (from 2.45% to 2.35%) and test-other (from 5.73% to 5.49%). This demonstrates that using a dual-output ASR model yields better performance for normalized output compared to normalizing a Punctuated-only ASR model.

Additionally, from the perspective of inference speed, all transducer-based ASR models achieve an RTF lower than 1, although they are higher than Whisper models. Notably, the proposed Conditioned Predictor ASR model has the lowest RTF, which is 0.35 lower (from 0.79 to 0.44) than the traditional cascaded system and 0.15 lower (from 0.59 to 0.44) than the 2-Decoder model, while maintaining similar performance. The Punctuated-only model has an RTF comparable to the Conditioned Predictor model, but this value will increase if normalization of the ASR output is required.

4.3.2. Original vs. auto-punctuated transcripts for training

Table 4 displays the test results on the CommonVoice and the AMI test sets. On out-of-domain test sets, our transducer-based models still outperform Whisper models, with a relative lower PC-WER up to 38% (from 73.27% to 45.24%).

In addition, by comparing Table 1 and Table 4, we observe some notable characteristics regarding the use of original versus auto-punctuated training data. Comparison of E2E ASR trained using original punctuated transcripts vs. auto-punctuated transcripts shows that the former yields lower punctuation and case error rates on the in-domain LibriSpeech test-clean and test-other sets, while the latter yields lower error rates on the out-of-domain CommonVoice and AMI test set. For instance, the 2-Decoder ASR model exhibits a PC-WER increase of 32% relative (from 8.85% to 11.67%) and 27% relative (from 11.76% to 14.88%) on LibriSpeech test-clean and test-other, respectively, when using auto-punctuated training transcripts (P'+N) instead of the original transcripts (P+N).

However, the same model exhibits a PC-WER reduction of 17% relative (from 35.79% to 29.70%) on the CommonVoice test set and 9% relative (from 50.18% to 45.43%) on the AMI test set when using auto-punctuated training transcripts. The differences mainly stem from the addition of case and punctuation. Models trained with auto-punctuated transcripts show a lower PuncER and CaseER. Specifically, for the Conditioned Predictor model on the CommonVoice test set, the PuncER reduction can be as high as 54% relative (from 60.35% to 27.44%), and the CaseER can be reduced by up to 51% relative (from 23.21% to 11.34%) when using auto-punctuated transcripts. This could be explained by the inconsistent (and sometimes erroneous) punctuation in LibriSpeech ground truth punctuated transcripts. By contrast, being trained on a wide range of domains, the LM outputs a better (possibly domain-dependent) punctuation than that learned on the specific LibriSpeech domain on average.

Table 5: Error rates (%) of Conditioned Predictor ASR (c) on LibriSpeech test-clean when using different proportions of punctuated and normalized (P + N) training data.

(P : N) Proportion	WER	PuncER	CaseER	PC-WER
0.50 : 0.50	**2.38**	29.33	**34.86**	**9.32**
0.35 : 0.65	**2.37**	**28.99**	45.44	9.95
0.20 : 0.80	**2.34**	29.96	56.37	10.83
0.05 : 0.95	2.56	31.59	58.58	11.46

Table 6: Error rates (%) of Conditioned Predictor ASR (c) on LibriSpeech test-clean for different scalar tradeoff parameters α when the proportion of punctuated data is 0.5.

Scalar ($\alpha : 1-\alpha$)	WER	PuncER	CaseER	PC-WER
No scalar	**2.38**	**29.33**	34.86	**9.32**
0.2 : 0.8	2.46	31.57	60.27	11.47
0.4 : 0.6	**2.39**	**29.47**	50.50	10.40
0.6 : 0.4	2.45	30.23	34.03	9.48
0.7 : 0.3	2.50	**29.26**	31.64	**9.21**
0.8 : 0.2	**2.43**	29.90	**30.24**	**9.15**
0.9 : 0.1	2.50	30.65	**30.47**	**9.35**

4.3.3. Conditioned Predictor ASR with limited punctuated data

Unlike the 2-Decoder ASR model, the Conditioned Predictor ASR model can be trained with different proportions of punctuated and normalized training data. To evaluate the effectiveness of this capability, we trained the Conditioned Predictor ASR model with different proportions of punctuated and normalized (P + N) training data. Table 5 presents the error rates obtained on the LibriSpeech test-clean set. It can be observed that the PC-WER and PuncER increase by small margins even when the amount of punctuated training data is reduced drastically. Using only 5% of punctuated training data can still achieve a PC-WER of 11.46%, i.e., a 2.42% absolute increase. This shows that the Predictor-Conditioned ASR model can effectively utilize normalized data to enhance the prediction of punctuation marks. Therefore, this model can serve scenarios having limited or severely limited punctuated training data.

Finally, we analyzed the influence of the tradeoff parameter α, while maintaining a fixed ratio of punctuated data to normalized data. Intuitively, an α value above 0.5 should achieve better punctuated transcription. But for the sake of rigor, we also tested α values of 0.2 and 0.4. Table 6 presents the results of this analysis. It shows that when the α value is 0.8, the model can achieve the best punctuated transcription performance, with a lowest PC-WER of 9.15%. With α above 0.5, even though the WER slightly increases, both the CaseER and the PuncER gradually decrease. Specifically, for the CaseER, there is a 50% relative reduction when the α value increases from 0.2 to 0.8. However, a high α value of 0.9 is not the optimal scenario. This could be attributed to the necessity of incorporating information from normalized training data to achieve a lower WER.

5. CONCLUSION

In this paper, we introduced two approaches to train a stateless transducer-based E2E joint punctuated and normalized ASR model with a limited amount of punctuated labeled data. The first approach uses a language model to generate auto-punctuated transcripts of normalized training data. Experimental results show that, compared to the model trained on punctuated data, the model trained on auto-punctuated data achieves a relative reduction in PC-WER of up to 17% on out-of-domain data. The second approach uses a single conditioned Predictor. Our proposed Conditioned Predictor model has a 42% lower relative PC-WER compared to the Whisper-base model, with a comparable number of parameters. For its normalized output on the LibriSpeech test sets, the proposed model achieves a 4% relative reduction in WER compared to the punctuated-only ASR model, highlighting the efficiency of having an ASR that can produce two types of outputs instead of solely punctuated output. Additionally, this approach demonstrates the feasibility of an E2E joint punctuated and normalized ASR system using as low as 5% punctuated training data with moderate (2.42% absolute) increase in errors. We also showed that scaling the punctuated and normalized data can result in a lower PC-WER and CaseER. Finally, as we use a stateless transducer-based architecture suitable for low-latency streaming applications, the proposed Conditioned Predictor ASR model has the lowest RTF among all the transducer-based ASR models, demonstrating its efficiency for streaming-ready applications.

6. ACKNOWLEDGMENTS

Experiments presented in this paper were carried out using (a) the Grid'5000 testbed, supported by a scientific interest group hosted by Inria and including CNRS, RENATER and several Universities as well as other organizations (see https://www.grid5000.fr), and (b) HPC resources from GENCI-IDRIS (Grant 2023-[AD011013881]).

7. REFERENCES

[1] Jonathan G Fiscus, Jerome Ajot, and John S Garofolo, "The rich transcription 2007 meeting recognition evaluation," in *International Evaluation Workshop on Rich Transcription*, 2007, pp. 373–389.

[2] Guillaume Gravier, Jean-François Bonastre, Edouard Geoffrois, Sylvain Galliano, Kevin McTait, and Khalid Choukri, "The ESTER evaluation campaign for the rich

transcription of french broadcast news.," in *International Conference on Language Resources and Evaluation (LREC)*, 2004.

[3] Ondřej Klejch, Peter Bell, and Steve Renals, "Punctuated transcription of multi-genre broadcasts using acoustic and lexical approaches," in *2016 IEEE Spoken Language Technology Workshop (SLT)*, 2016, pp. 433–440.

[4] Binh Nguyen, Vu Bao Hung Nguyen, Hien Nguyen, Pham Ngoc Phuong, The-Loc Nguyen, Quoc Truong Do, and Luong Chi Mai, "Fast and accurate capitalization and punctuation for automatic speech recognition using transformer and chunk merging," in *2019 22nd conference of the oriental COCOSDA international committee for the co-ordination and standardisation of speech databases and assessment techniques (O-COCOSDA)*, 2019, pp. 1–5.

[5] Junwei Liao, Sefik Eskimez, Liyang Lu, Yu Shi, Ming Gong, Linjun Shou, Hong Qu, and Michael Zeng, "Improving readability for automatic speech recognition transcription," *ACM Transactions on Asian and Low-Resource Language Information Processing*, vol. 22, no. 5, pp. 1–23, 2023.

[6] Jiangyan Yi, Jianhua Tao, Ye Bai, Zhengkun Tian, and Cunhang Fan, "Adversarial transfer learning for punctuation restoration," *arXiv preprint arXiv:2004.00248*, 2020.

[7] Qiushi Huang, Tom Ko, H Lilian Tang, Xubo Liu, and Bo Wu, "Token-level supervised contrastive learning for punctuation restoration," in *INTERSPEECH 2021*, 2021, pp. 2012–2016.

[8] Martin Poláček, Petr Červa, Jindřich Žďánský, and Lenka Weingartová, "Online punctuation restoration using ELECTRA model for streaming asr systems," in *INTERSPEECH 2023*, 2023, pp. 446–450.

[9] Mohammadreza Ghodsi, Xiaofeng Liu, James Apfel, Rodrigo Cabrera, and Eugene Weinstein, "RNN-transducer with stateless prediction network," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 7049–7053.

[10] Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning, "ELECTRA: Pre-training text encoders as discriminators rather than generators," in *International Conference on Learning Representations (ICLR)*, 2020.

[11] Hayato Futami, Emiru Tsunoo, Kentaro Shibata, Yosuke Kashiwagi, Takao Okuda, Siddhant Arora, and Shinji Watanabe, "Streaming joint speech recognition and disfluency detection," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2023, pp. 1–5.

[12] Tomohiro Tanaka, Ryo Masumura, Mana Ihori, Akihiko Takashima, Shota Orihashi, and Naoki Makishima, "End-to-end rich transcription-style automatic speech recognition with semi-supervised learning," in *INTERSPEECH 2021*, 2021, pp. 4458–4462.

[13] Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever, "Robust speech recognition via large-scale weak supervision," in *International Conference on Machine Learning*, 2023, pp. 28492–28518.

[14] Jumon Nozaki, Tatsuya Kawahara, Kenkichi Ishizuka, and Taiichi Hashimoto, "End-to-end speech-to-punctuated-text recognition," in *INTERSPEECH 2022*, 2022, pp. 1811–1815.

[15] Mana Ihori, Hiroshi Sato, Tomohiro Tanaka, Ryo Masumura, Saki Mizuno, and Nobukatsu Hojo, "Transcribing speech as spoken and written dual text using an autoregressive model," in *INTERSPEECH 2023*, 2023, pp. 461–465.

[16] Cal Peyser, Hao Zhang, Tara N Sainath, and Zelin Wu, "Improving performance of end-to-end ASR on numeric sequences," *arXiv preprint arXiv:1907.01372*, 2019.

[17] Monica Sunkara, Chaitanya Shivade, Sravan Bodapati, and Katrin Kirchhoff, "Neural inverse text normalization," in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021, pp. 7573–7577.

[18] Ineke Schuurman, Machteld Schouppe, Heleen Hoekstra, and Ton Van der Wouden, "CGN, an annotated corpus of spoken dutch," in *4th International Workshop on Linguistically Interpreted Corpora (LINC-03)*, 2003.

[19] Rosana Ardila, Megan Branson, Kelly Davis, Michael Henretty, Michael Kohler, Josh Meyer, Reuben Morais, Lindsay Saunders, Francis M Tyers, and Gregor Weber, "Common voice: A massively-multilingual speech corpus," *arXiv preprint arXiv:1912.06670*, 2019.

[20] Vineel Pratap, Qiantong Xu, Anuroop Sriram, Gabriel Synnaeve, and Ronan Collobert, "MLS: A large-scale multilingual dataset for speech research," in *Interspeech*, 2020, pp. 2757–2761.

[21] Monica Sunkara, Srikanth Ronanki, Kalpit Dixit, Sravan Bodapati, and Katrin Kirchhoff, "Robust prediction of punctuation and truecasing for medical ASR," in *Workshop on Natural Language Processing for Medical Conversations*, 2020, pp. 53–62.

[22] Alex Graves, "Sequence transduction with recurrent neural networks," *arXiv preprint arXiv:1211.3711*, 2012.

[23] Jinyu Li, "Recent advances in end-to-end automatic speech recognition," *APSIPA Transactions on Signal and Information Processing*, vol. 11, no. 1, 2022.

[24] Shubham Toshniwal, Tara N Sainath, Ron J Weiss, Bo Li, Pedro Moreno, Eugene Weinstein, and Kanishka Rao, "Multilingual speech recognition with a single end-to-end model," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018, pp. 4904–4908.

[25] Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur, "Librispeech: An ASR corpus based on public domain audio books," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2015, pp. 5206–5210.

[26] Aleksandr Meister, Matvei Novikov, Nikolay Karpov, Evelina Bakhturina, Vitaly Lavrukhin, and Boris Ginsburg, "LibriSpeech-PC: Benchmark for evaluation of punctuation and capitalization capabilities of end-to-end asr models," in *2023 Automatic Speech Recognition and Understanding Workshop (ASRU)*, 2023, pp. 1–7.

[27] R. Ardila, M. Branson, K. Davis, M. Henretty, M. Kohler, J. Meyer, R. Morais, L. Saunders, F. M. Tyers, and G. Weber, "Common voice: A massively-multilingual speech corpus," in *Proceedings of the 12th Conference on Language Resources and Evaluation (LREC 2020)*, 2020, pp. 4211–4215.

[28] Jean Carletta, Simone Ashby, Sebastien Bourban, Mike Flynn, Mael Guillemot, Thomas Hain, Jaroslav Kadlec, Vasilis Karaiskos, Wessel Kraaij, Melissa Kronenthal, et al., "The AMI meeting corpus: A pre-announcement," in *International Workshop on Machine Learning for Multimodal Interaction*, 2005, pp. 28–39.

[29] NIST, "SCTK," `https://github.com/usnistgov/SCTK.git`, 2024, GitHub repository.

[30] Taku Kudo and John Richardson, "Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing," in *2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, 2018, pp. 66–71.

[31] Zengwei Yao, Liyong Guo, Xiaoyu Yang, Wei Kang, Fangjun Kuang, Yifan Yang, Zengrui Jin, Long Lin, and Daniel Povey, "Zipformer: A faster and better encoder for automatic speech recognition," *arXiv preprint arXiv:2310.11230*, 2023.

[32] Daulet Nurmanbetov, "rpunct," `https://github.com/Felflare/rpunct.git`, 2021, GitHub repository.

[33] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT)*, 2019, vol. 1, p. 2.