Pack&Shift: Efficient RNNT Training via Sequence Packing and Chunked Attention with Shifted Windows

Anonymous ACL submission

Abstract

Training Recurrent Neural Network Transducer (RNNT) with self-attention encoders is increasingly bottlenecked by wasted computation on zero padding and the quadratic cost of long-context attention. We propose Pack&Shift, an efficient training pipeline that combines (i) sequence packing to eliminate intra-batch padding, (ii) chunked attention to cap attention context, and (iii) shifted windows to mitigate the accuracy degradation commonly observed at chunk boundaries. Pack&Shift packs variable-length utterances into a single contiguous sequence, applies convolutional subsampling and chunked self-attention in a compute-efficient manner, and then unpacks encoder outputs back to utterance granularity for standard RNNT loss calculation. Experiments on LibriSpeech and VoxPopuli with a Conformer-based RNNT show that Pack&Shift reduces the padding ratio from 30–60% to around 1%, lowers GPU memory usage by about 40%, and improves training throughput by about 12-24% while maintaining comparable recognition performance to conventional padded training.

Keywords: speech recognition, RNNT, sequence packing, chunking, shifted window, efficiency

1 Introduction

Neural network–based automatic speech recognition (ASR) models are trained using gradients computed over mini-batches. Since speech utterances vary in length, shorter samples are commonly zero-padded to match the longest utterance in the batch. Although simple and broadly applicable, this forces the model to compute over non-informative padding regions. The resulting computational and memory waste becomes especially severe in large-scale training or length-dispersed corpora, limiting feasible batch size and effective context length.

In natural language processing (NLP), sequence packing is routinely adopted as a standard technique to mitigate padding inefficiency by concatenating variable-length sequences into a single longer sequence (Krell et al., 2021; Wang et al., 2025). In contrast, ASR takes continuous frame sequences as input and typically lacks deterministic input–output alignment information, which makes it difficult to directly apply packing techniques from NLP; accordingly, related studies remain relatively limited.

In this work, we propose a training pipeline that combines sequence packing, chunking, and shifted windows for efficient ASR training. Sequence packing removes most padding-induced waste, reducing the padding ratio from 30–60% to approximately 1%. Chunking further bounds the quadratic cost of self-attention, while shifted windows alleviate the recognition degradation often observed at chunk boundaries (Liu, 2021).

Our experiments on LibriSpeech show that Pack&Shift reduces GPU memory usage by approximately 40% and improves training speed by about 12% over conventional zero-padding–based training. In LibriSpeech ablations, shifted windows reduce word error rate (WER) by 3.2-15.9% relative compared with fixed chunk boundaries, depending on chunk size and evaluation split. Overall, Pack&Shift achieves WER comparable to the zero-padding baseline, while requiring less memory and lower latency at the same batch size.

We summarize our contributions as follows:

1. We propose a training pipeline that integrates sequence packing, chunking, and unpacking to prevent the computational and memory waste caused by zero padding in RNNT.

2. We introduce shifted windows to mitigate the performance degradation of chunked attention while minimizing additional computational overhead.

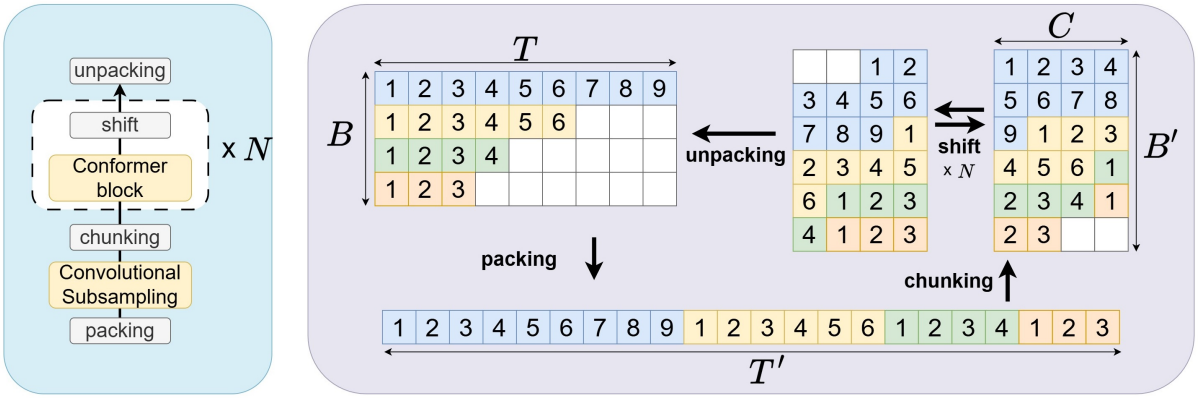

Figure 1: Diagram of the proposed Pack&Shift-based training pipeline

2 Proposed Method

2.1 Sequence Packing for ASR

We propose a training pipeline that removes in-batch padding through sequence packing, bounds self-attention cost through chunking, and restores utterance-level representations through unpacking. The overall procedure is illustrated in Figure 1.

Let $X \in \mathbb{R}^{B\times T\times F}$ denote a mini-batch of B speech feature sequences, where T is the maximum length in the batch and F is the feature dimension. A Conformer encoder consists of a convolutional subsampling layer followed by N Conformer modules (Gulati et al., 2020). The subsampling layer reduces the time length by a factor of $1/R$; for simplicity, we omit this reduction in the notation below, while using $R = 8$ in our experiments.

Stage 1: Sequence packing. We first remove padded regions by concatenating only valid frames in the batch: $\mathbb{R}^{B\times T\times F} \rightarrow \mathbb{R}^{1\times T'\times F}$, where T' is the total number of valid frames.

Stage 2: Convolutional subsampling. The packed sequence is fed into the CNN subsampling layer. Since $T' \leq BT$, packing reduces computation and memory usage in this stage.

Stage 3: Chunking. Directly applying self-attention to the packed sequence can be expensive because its cost grows quadratically with sequence length. We therefore split the packed sequence into chunks of length C: $\mathbb{R}^{1\times T'\times F} \rightarrow \mathbb{R}^{B'\times C\times F}$, where B' is the number of chunks, $C = \min(C_{\max}, T)$ and $C_{\max}$ is a tuning factor. If necessary, we add padding so that T' is divisible by C; the additional padding ratio is upper-bounded by $1/B'$.

Stage 4: Conformer modules. The chunked features are passed through N Conformer modules. After packing and chunking, $B'C \leq BT$, and the attention context is bounded by C rather than T. Thus, the proposed representation reduces padding-related computation while controlling the self-attention memory bottleneck.

Stage 5: Unpacking. Finally, we unpack the encoder outputs back to utterance-level sequences using the original length information. The recovered representations are then used to compute the standard utterance-level RNNT loss.

2.2 Shifted Window

Although chunking reduces the cost of self-attention, fixed chunk boundaries can degrade recognition performance by limiting context near boundary regions (Dong et al., 2019; Wu et al., 2020). To mitigate this issue, we adopt the shifted-window mechanism of (Liu, 2021), shifting chunk boundaries by shift size $= C/2$ at each Conformer layer. This redefines the attention boundaries across layers, allowing neighboring chunks to exchange contextual information. Since the operation can be implemented with lightweight tensor operations such as reshaping and rolling, it introduces little overhead.

3 Related Works

Padding inefficiency in speech processing has mainly been addressed through length bucketing, dynamic batching, or sorted batching (Variani et al., 2017; Ge et al., 2021; Gonzalez et al., 2023). These methods reduce padding by grouping utterances with similar lengths, and have been studied in text-to-speech, speech enhancement, and encoder-decoder ASR settings (Ge et al., 2021; Gonzalez et al., 2023; Żelasko et al., 2025). However, they still retain zero padding within each batch and re-

Table 1: Comparison of padding ratio.

	Padding ratio (%)	
	Libri	Vox
Baseline	28.3	61.3
Bucketing	15.2	24.6
Pack&Shift	**0.6**	**1.4**

quire careful tuning of bins or batch sizes.

Sequence packing has recently gained attention in speech processing, especially with the emergence of speech language models (SLMs). For example, (Zhang et al., 2025) concatenates multiple audio segments into a single sequence and masks cross-segment attention. However, naive packing for ASR can lead to sparse attention and RNNT loss computation with high masking ratios, causing substantial memory and computational waste.

Chunked attention has been widely studied for streaming ASR and long-form transcription (Chiu and Raffel, 2018; Le et al., 2025). A common way to reduce chunk-boundary degradation is to introduce overlapping chunks (Dong et al., 2019), but this increases computation. Shifted-window mechanisms have also been adapted to ASR to improve context modeling across chunk boundaries with limited overhead (Wang and Xu, 2022; Liu, 2021). Nevertheless, prior shifted-chunk approaches still rely on zero padding, leaving padding-induced inefficiency unresolved.

4 Experiments

4.1 Data

We evaluate on LibriSpeech and six-language VoxPopuli (de, en, es, fr, it, pl) (Panayotov et al., 2015; Wang et al., 2021). Speech inputs are 80-dimensional log-Mel filterbank features extracted with a 25 ms window and 10 ms frame shift. We apply SpecAugment during training (Park et al., 2019), and use BPE tokenization with a vocabulary size of 3,000 for each dataset (Kudo and Richardson, 2018).

4.2 Model

We use a Conformer-based RNNT model (Graves, 2012; Gulati et al., 2020). The encoder has 17 Conformer layers with model dimension 512 and 8 attention heads. A three-layer CNN front-end with 256 channels performs 1/8 subsampling, and the Conformer convolution module uses kernel size 31 with batch normalization. The prediction network is a 1-layer LSTM, and the joint network is a 2-layer feed-forward network.

Table 2: Comparison of the batch size (B), training speed (step time), and throughput (utterances per second) across datasets.

	B (avg)		step time		Utt/sec	
	Libri	Vox	Libri	Vox	Libri	Vox
Baseline	72.0	52.0	1.14	1.20	63.2	43.3
Pack&Shift	72.0	52.0	**1.00**	**0.91**	72.0	57.1
	120.0	80.0	1.49	1.28	**80.5**	62.5
Bucketing	91.8	109.8	1.17	1.45	78.5	**75.7**

Models are trained for 50,000 steps using Adam on 4 NVIDIA A100 GPUs, with gradient accumulation set to 2. To avoid RNNT-loss memory spikes, we compute the loss over length-sorted micro-batches of 4 utterances.

4.3 Evaluation & Implementation Details

We validate every 1,000 steps and report the checkpoint with the lowest validation WER. All results use greedy decoding without external language models. Bucketing uses a bin size of 4 with manually tuned per-bin batch sizes. Pack&Shift uses $C_{\max} = 500$ unless otherwise specified, sets the shift size to $C/2$, and restricts C to even values. Cross-utterance masks, when used, are also shifted consistently with the features.

5 Results

We conduct experiments on two benchmarks with different duration characteristics. LibriSpeech is an English dataset recorded in studio conditions, whose duration distribution is skewed toward longer utterances. VoxPopuli is a multilingual dataset collected from real meeting environments; its duration distribution is skewed toward shorter utterances, while also containing a small number of exceptionally long outliers.

We first analyze the proportion of feature frames occupied by zero padding and report the results in Table 1. The baseline contains 28-61% padding, whereas Pack&Shift reduces the padding ratio to 0.6-1.4%. This is substantially lower than bucketing, which still leaves 15-25% padding. Unlike bucketing, Pack&Shift does not require increasing the number of bins or manually tuning the batch size for each bin.

Next, we compare the baseline and Pack&Shift under the same batch size. Table 2 shows that Pack&Shift improves training speed by 12-24%.

Table 3: Comparison of training efficiency (batch size/memory) and recognition performance (WER) under different Pack&Shift configurations on LibriSpeech. Here, Shift denotes the shifted-window scheme, and Mask indicates whether cross-utterance masking is applied.

Method	Batch size (B, avg)	Memory Usage (peak)	$C_{\max}$	Shift	Mask	WER (%) test clean	test other	dev clean	dev other
Baseline	72.0	∼80 GB	-	-	-	3.04	7.63	2.83	7.72
Bucketing	91.8	∼80 GB	-	-	-	2.99	7.24	2.70	7.32
Pack&Shift	72.0	**∼47 GB**	500 (40 sec)	O	O	3.06	7.60	2.89	7.69
	120.0	∼80 GB	100 (8 sec)	X	O	3.94	9.35	3.65	9.67
			100 (8 sec)	O	O	3.36	8.38	3.07	8.62
			300 (24 sec)	X	O	3.38	7.60	3.11	7.83
			300 (24 sec)	O	O	3.12	7.23	2.80	7.58
			500 (40 sec)	X	O	3.11	7.53	2.91	7.56
			500 (40 sec)	O	X	**2.80**	**6.82**	**2.56**	**7.03**
			500 (40 sec)	O	O	2.84	6.84	2.58	7.03

Table 4: Comparison of Pack&Shift performance (WER) in the VoxPopuli multilingual setting.

Method	Batch size (B, avg)	Memory Usage (peak)	$C_{\max}$	Shift	Mask	WER (%) avg	De	En	Es	Fr	It	Pl
Baseline	52.0	∼80 GB	-	-	-	14.18	13.91	9.96	11.75	12.92	22.72	13.81
Bucketing	109.0	∼80 GB	-	-	-	13.68	14.26	10.10	11.34	12.69	**19.89**	13.78
Pack&Shift	52.0	**∼55 GB**	500 (40 sec)	O	O	14.31	14.22	10.06	12.10	13.38	21.55	14.52
	80.0	∼80 GB		O	X	**13.44**	**13.43**	**9.65**	**11.21**	**12.37**	20.64	**13.34**
	80.0	∼80 GB		O	O	13.70	13.52	9.96	11.37	12.85	21.03	13.49

As shown in Table 3 and Table 4, Pack&Shift reduces peak GPU memory usage by approximately 30-40% compared to the zero-padding baseline. In terms of WER, Pack&Shift shows no meaningful degradation on LibriSpeech, but yields a slight degradation on VoxPopuli. We attribute this gap to the reduced effective context size introduced by chunking, which can be more influential in multilingual real-world speech.

The memory savings of Pack&Shift also allow us to increase the batch size up to the practical GPU memory limit. As reported in Table 3 and Table 4, the feasible batch size increases by 67% on LibriSpeech and by 54% on VoxPopuli. In these maximum-batch experiments, the performance gain from larger batch sizes outweighs the loss caused by restricted context. Similar gains are observed for bucketing, but Pack&Shift shows larger improvements.

On VoxPopuli, the bucketing approach still underperforms Pack&Shift even with a larger effective batch size. We also observed larger performance variance across languages under bucketing than under the baseline or Pack&Shift. This suggests that duration-based bucketing may interact with language, recording environment, or utterance-duration imbalance in VoxPopuli, which can introduce bias during stochastic gradient descent and degrade multilingual performance.

Finally, Table 3 reports the ablation results. We compare Pack&Shift with different chunk sizes ($C_{\max}$), with or without shifted windows, and with or without cross-utterance masking. Increasing the chunk size consistently improves WER. Removing shifted windows causes a 3-10% relative degradation, indicating that shifted boundaries are important for mitigating chunk-boundary effects. Interestingly, removing cross-utterance masking improves WER across all settings, especially on VoxPopuli. We interpret this as a regularization effect: cross-utterance interaction may improve robustness more than it harms training through information leakage.

6 Conclusion

We proposed Pack&Shift, a training pipeline that combines sequence packing, chunked attention, and shifted windows to reduce both zero-padding waste and long-sequence self-attention cost in RNNT training. Pack&Shift reduces the padding ratio from 30-60% to 1%, lowers GPU memory usage by about 40%, and improves training throughput by 12-24%, while maintaining WER comparable to conventional zero-padding-based training.

Limitations

This work has several limitations. First, our experiments are limited to Conformer-based RNNT models, although Pack&Shift is designed to be applicable to other encoder-based ASR architectures. Second, we evaluate the method on LibriSpeech and six-language VoxPopuli only, so further validation is needed on larger-scale, noisier, and more diverse speech corpora. Third, all results are obtained with greedy decoding and without an external language model, leaving the interaction between Pack&Shift and more advanced decoding strategies unexplored. Finally, Pack&Shift introduces additional implementation complexity for packing, chunking, shifting, and unpacking, which may require careful engineering in distributed training setups.

References

Chung-Cheng Chiu and Colin Raffel. 2018. Monotonic chunkwise attention. In *International Conference on Learning Representations*.

Linhao Dong, Feng Wang, and Bo Xu. 2019. Self-attention aligner: A latency-control end-to-end model for asr using self-attention network and chunk-hopping. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5656–5660. IEEE.

Zhenhao Ge, Lakshmish Kaushik, Masanori Omote, and Saket Kumar. 2021. Speed up training with variable length inputs by efficient batching strategies. In *Proc. Interspeech 2021*, pages 156–160.

Philippe Gonzalez, Tommy Sonne Alstrøm, and Tobias May. 2023. On batching variable size inputs for training end-to-end speech enhancement systems. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE.

Alex Graves. 2012. Sequence Transduction with Recurrent Neural Networks. In *International Conference on Machine Learning (ICML) Workshop on Representation Learning*.

Anmol Gulati, James Qin, Chung-Cheng Chiu, Niki Parmar, Yu Zhang, Jiahui Yu, Wei Han, Shibo Wang, Zhengdong Zhang, Yonghui Wu, and 1 others. 2020. Conformer: Convolution-augmented transformer for speech recognition. *Interspeech 2020*.

Mario Michael Krell, Matej Kosec, Sergio P Perez, and Andrew Fitzgibbon. 2021. Efficient sequence packing without cross-contamination: Accelerating large language models without impacting performance. *arXiv preprint arXiv:2107.02027*.

Taku Kudo and John Richardson. 2018. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 conference on empirical methods in natural language processing: System demonstrations*, pages 66–71.

Khanh Le, Tuan Vu Ho, Dung Tran, and Duc Thanh Chau. 2025. Chunkformer: Masked chunking conformer for long-form speech transcription. In *ICASSP 2025-2025 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE.

Z Liu. 2021. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, page 10012.

Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. 2015. Librispeech: an asr corpus based on public domain audio books. In *2015 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 5206–5210. IEEE.

Daniel S Park, William Chan, Yu Zhang, Chung-Cheng Chiu, Barret Zoph, Ekin D Cubuk, and Quoc V Le. 2019. Specaugment: A simple data augmentation method for automatic speech recognition. *Interspeech 2019*, page 2613.

Ehsan Variani, Tom Bagby, Erik McDermott, and Michiel Bacchiani. 2017. End-to-end training of acoustic models for large vocabulary continuous speech recognition with tensorflow. In *Interspeech*, pages 1641–1645.

Changhan Wang, Morgane Rivière, Ann Lee, Anne Wu, Chaitanya Talnikar, Daniel Haziza, Mary Williamson, Juan Pino, and Emmanuel Dupoux. 2021. Voxpopuli: A large-scale multilingual speech corpus for representation learning, semi-supervised learning and interpretation. In *ACL 2021-59th Annual Meeting of the Association for Computational Linguistics*.

Fangyuan Wang and Bo Xu. 2022. Shifted chunk encoder for transformer based streaming end-to-end asr. In *International Conference on Neural Information Processing*, pages 39–50. Springer.

Shuhe Wang, Guoyin Wang, Yizhong Wang, Jiwei Li, Eduard Hovy, and Chen Guo. 2025. Packing analysis: Packing is more appropriate for large models or datasets in supervised fine-tuning. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 4953–4967.

Chunyang Wu, Yongqiang Wang, Yangyang Shi, Ching-Feng Yeh, and Frank Zhang. 2020. Streaming transformer-based acoustic models using self-attention with augmented memory. In *Proc. Interspeech 2020*, pages 2132–2136.

Piotr Żelasko, Kunal Dhawan, Daniel Galvez, Krishna C Puvvada, Ankita Pasad, Nithin Rao Koluguri, Ke Hu,

Vitaly Lavrukhin, Jagadeesh Balam, and Boris Ginsburg. 2025. Training and inference efficiency of encoder-decoder speech models. *arXiv preprint arXiv:2503.05931*.

Binbin Zhang, Chengdong Liang, Shuai Wang, Xuelong Geng, Zhao Guo, Haoyu Li, Hao Yin, Xipeng Yang, Pengshen Zhang, Changwei Ma, and 1 others. 2025. West: Llm based speech toolkit for speech understanding, generation, and interaction. *arXiv preprint arXiv:2509.19902*.