Squeezeformer: An Efficient Transformer for Automatic Speech Recognition

Anonymous Author(s)
Affiliation
Address
`email`

Abstract

The recently proposed Conformer model has become the *de facto* backbone model for various downstream speech tasks based on its hybrid attention-convolution architecture that captures both local and global features. However, through a series of systematic studies, we find that the Conformer architecture's design choices are not optimal. After reexamining the design choices for both the macro and micro-architecture of Conformer, we propose Squeezeformer which consistently outperforms the state-of-the-art ASR models under the same training schemes. In particular, for the macro-architecture, Squeezeformer incorporates (i) the Temporal U-Net structure which reduces the cost of the multi-head attention modules on long sequences, and (ii) a simpler block structure of feed-forward module followed up by multi-head attention or convolution modules instead of the Macaron structure proposed in Conformer. Furthermore, for the micro-architecture, Squeezeformer (i) simplifies the activations in the convolutional block, (ii) removes redundant Layer Normalization operations, and (iii) incorporates an efficient depth-wise downsampling layer to efficiently sub-sample the input signal. Squeezeformer achieves state-of-the-art results of 7.5%, 6.5%, and 6.0% word-error-rate (WER) on Librispeech test-other without external language models, which are 3.1%, 1.4%, and 0.6% better than Conformer-CTC with the same number of FLOPs. Our code is open sourced and available online [1].

1 Introduction

The increasing success of end-to-end neural network models has been a huge driving force for the drastic advancements in various automatic speech recognition (ASR) tasks. While both convolutional neural networks (CNN) [17, 24, 26, 33] and Transformers [22, 28, 29, 53, 54] have drawn attention as popular backbone architectures for ASR models, each of them has several limitations. Generally, CNN models lack the ability to capture global contexts and Transformers involve prohibitive computing and memory overhead. To overcome these shortcomings, Conformer [14] has recently proposed a novel convolution-augmented Transformer architecture. Due to its ability to synchronously capture global and local features from audio signals, Conformer has become the *de facto* model not only for ASR tasks, but also for various end-to-end speech processing tasks [15]. Furthermore, it has also achieved the state-of-the-art performance in combination with recent developments in semi-supervised learning methodologies as well [34, 57].

Despite being a key architecture in speech processing tasks, the Conformer architecture has some limitations that can be improved upon. First, Conformer still suffers from the quadratic complexity of the attention mechanism limiting it's efficiency on long sequence lengths This problem is further

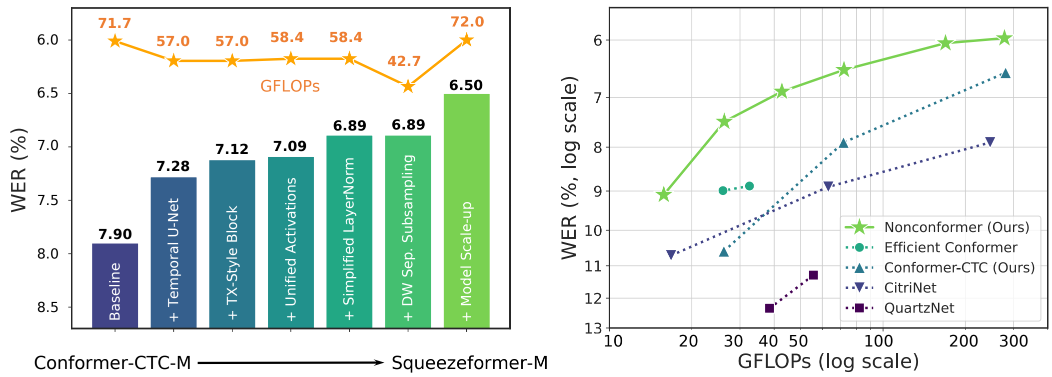

Figure 1: (Left) We perform a series of systematic studies on macro and micro architecture to redesign the Conformer architecture towards our Squeezeformer architecture. The bars and the line indicate the WER on Librispeech test-other dataset and the FLOPs, respectively. For each design modification, we strictly improve WER until our final Squeezeformer model outperforms Conformer by 1.40% WER improvement with the same number of FLOPs. See Tab. 1 for the details. (Right) Librispeech test-other WER vs. FLOPs for Squeezeformer and other state-of-the-art ASR models. Our architecture scales well to smaller and larger models to constantly outperform other models by a large margin throughout the entire FLOPs range. For both plots, the lower the WER, the better; however, we plotted in reverse for better visualization.

highlighted by the long sequence lengths of typical audio inputs as alow pointed out in [43]. Furthermore, the Conformer architecture is relatively more complicated than Transformer architectures used in other domains such as in natural language processing [8, 41, 48] or computer vision [10, 11, 46]. For instance, the Conformer architecture incorporates multiple different normalization schemes and activation functions, the Macaron structure [31], as well as back-to-back multi-head attention (MHA) and convolution modules. This level of complexity makes it difficult to efficiently deploy the model on dedicated hardware platforms for inference [23, 36, 52]. More importantly, this raises the question of whether such design choices are necessary and optimal for achieving good performance in ASR tasks.

In this paper, we perform a careful and systematic analysis of each of the design choices with the goal of achieving lower word-error-rate (WER) for a given computational budget. We developed a much simpler and more efficient hybrid attention-convolution architecture in both its macro and micro-design that consistently outperforms the state-of-the-art ASR models. In particular, we make the following contributions in our proposed Squeezeformer model:

- We find a high temporal redundancy in the learned feature representations of neighboring speech frames especially deeper in the network which results in unnecessary computational overhead. To address this, we incorporate the temporal U-Net structure in which a downsampling layer doubles the sampling rate at the middle of the network, and a light upsampling layer recovers the temporal resolution at the end for training stability (§ 3.1.1).

- We redesign the hybrid attention-convolution architecture based on our observation that the back-to-back MHA and convolution modules with the Macaron structure is suboptimal. In particular, we propose a simpler block structure similar to the standard Transformer block [8, 48], where the MHA and convolution modules are each directly followed by a single feed forward module (§ 3.1.2).

- We finely examined the micro-architecture of the network and found several modifications that simplify the model overall and greatly improve the accuracy and efficiency. This includes (i) activation unification that replaces GLU activations with Swish (§ 3.2.1), (ii) Layer Normalization simplification by replacing redundant pre-Layer Normalization layers with a Scaled Post-LN which incorporates a learnable scaling for the residual path which can be merged with other layers to be zero-cost during inference (§ 3.2.2), and (iii) incorporation of a depth-wise separable convolution for the first sub-sampling layer that results in a significant FLOPs reduction (§ 3.2.3).

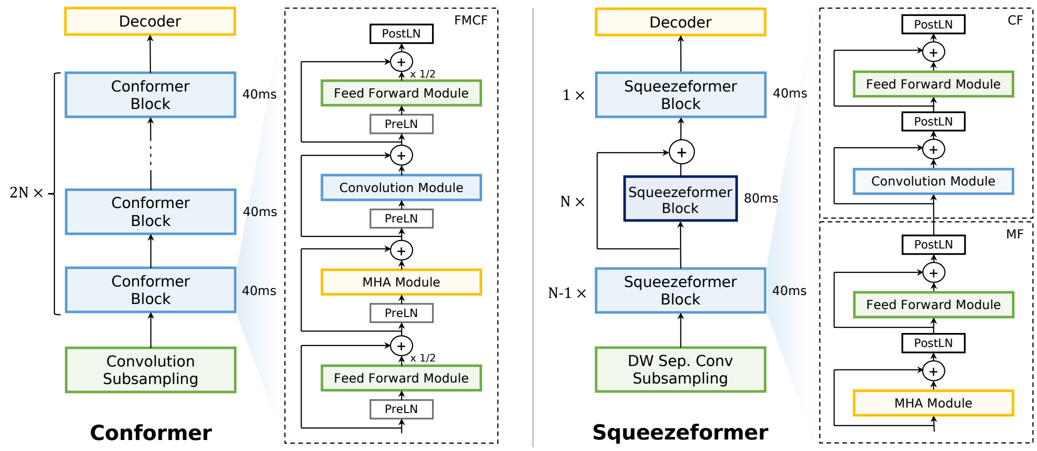

Figure 2: The Conformer architecutre (Left) and the Squeezeformer architecture (Right) which comprises of the Temporal U-Net structure for downsampling and upsampling of the sampling rate, the standard Transformer-style block structure that only uses Post-Layer Normalization, and the depth-wise separable subsampling layer.

- We show that the Squeezeformer architecture scales well with both smaller and larger models and consistently outperforms other state-of-the-art ASR models when trained under the same settings (Tab. 4.2, § 4.2). Furthermore, we justify the final model architecture of Squeezeformer with a reverse ablation study for the design choices (Tab. A.1, § 4.3).

2 Related Work

The recent advancements in end-to-end ASR can be broadly categorized into (1) model architecture and (2) training methods.

Model Architecture for End-to-end ASR The recent end-to-end ASR models are typically composed of an *encoder*, which takes as input a speech signal (i.e., sequence of speech frames) and extracts high-level acoustic features, and a *decoder*, which converts the extracted features from the encoder into a sequence of text. The model architecture of the encoder determines the representational power of an ASR model and its ability to extract acoustic features from input signals. A strong architecture is critical for overall performance.

One of the popular choices for a backbone model architecture is CNN. End-to-end deep CNN models have been first explored in Jasper [26], and further improved by introducing depthwise separable convolution [19, 44, 47] in QuartzNet [24] and the Squeeze-and-Excitation module [21] in CitriNet [33] and ContextNet [17]. However, since CNN often fails to capture global contexts, Transformer [48] models have also been widely adopted in backbone architectures due to their ability to capture long-range dependencies between speech frames [22, 28, 29, 53, 54]. Recently, [14] has proposed a novel model architecture named Conformer, which augments Transformers with convolutions to model both global and local dependencies efficiently. In this work, with the Conformer architecture as our starting point, we focus on designing a next-generation model architecture for ASR that is simpler, more accurate, and more efficient.

The hybrid attention-convolution architecture of Conformer has enabled the state-of-the-art results in many speech tasks. However, the quadratic complexity of the attention layer still proves to be cost prohibitive at larger sequence lengths. While different approaches have been proposed to reduce the cost of MHA in ASR [6, 43, 55, 56], their main focus is not changing the overall architecture design and their optimizations can also be applied orthogonal to Squeezeformer. Efficient-Conformer [5] introduces the progressive downsampling scheme and grouped attention to reduce the training and inference costs of Conformer. Our work incorporates a similar progressive downsampling, but also introduces an up-sampling mechanism with skip connections from the earlier layers inspired by the U-Net [42] architecture in computer vision and U-Time [40] architecture for sleep signal analysis. We

Table 1: Starting from Conformer as the baseline, we redesign the architecture towards Squeezeformer through a series of systematic studies on macro and micro architecture. Note that for each design change, the WER on LibriSpeech test-clean and test-other datasets improves consistently. For comparison, we include the number of parameters and FLOPs for a 30s input in the last two columns.

Model	Design change	test-clean	test-other	Params (M)	GFLOPs
Conformer-CTC-M	Baseline	3.20	7.90	27.4	71.7
	+ Temporal U-Net (§ 3.1.1)	2.97	7.28	27.5	57.0
	+ Transformer-style Block (§ 3.1.2)	2.93	7.12	27.5	57.0
	+ Unified activations (§ 3.2.1)	2.88	7.09	28.7	58.4
	+ Simplified LayerNorm (§ 3.2.2)	2.85	6.89	28.7	58.4
Squeezeformer-SM	+ DW sep. subsampling (§ 3.2.3)	2.79	6.89	28.2	42.7
Squeezeformer-M	+ Model scale-up (§ 3.2.3)	2.56	6.50	55.6	72.0

find this to be critical for training stability and overall performance. In addition, through systematic experiments, we completely refactor the Conformer block by carefully redesigning both the macro and micro-architectures.

Training Methodology for End-to-end ASR. In the past few years, various self-supervised learning methodologies based on contrastive learning [3, 49, 57] or masked prediction [2, 7, 20] have been proposed to push forward the ASR performance. While a model pre-trained with a self-supervised task generally outperforms when finetuned on a target ASR task, training strategies are not the main focus in this work as they can be applied independently to the underlying architecture.

3 Architecture Design

The Conformer architecture has been widely adopted by the speech community and is used as a backbone for different speech tasks. At a macro-level, Conformer incorporates the Macaron structure [31] comprised of four modules per block, as shown in Fig. 2 (Left). These blocks are stacked multiple times to construct the Conformer architecture. In this work, we carefully reexamine the design choices in Conformer, starting first with its macro-architecture, and then its micro-architecture design. We choose Conformer-CTC-M as the baseline model for the case study, and we compare word-error-rate (WER) on Librispeech test-other and FLOPs on a 30s audio input as performance metrics for each architecture. Furthermore, we focus on FLOPs as a proxy for model efficiency. While we acknowledge that FLOPs may not always be a linear indicator of hardware efficiency/runtime, we choose FLOPs as it is hardware agnostic and is statically computable. However, we do measure the final latency of our changes, ensuring up to 30% consistent improvement in runtime for different versions of Squeezeformer (Tab. 4.2)

3.1 Macro-Architecture Design

We first focus on designing the macro structure of Squeezeformer, i.e., how the blocks and modules are organized in a global scale.

3.1.1 Temporal U-Net Architecture

The hybrid attention-convolution structure enables Conformer to capture both global and local interactions. However, the attention operation has a quadratic FLOP complexity with respect to the input sequence length. We propose to lighten this extra overhead by computing attention over a reduced sequence length. In the Conformer model itself, the input sampling rate is reduced from 10ms to 40ms with a convolutional subsampling block at the base of the network. However, this rate is kept constant throughout the network, with all the attention and convolution operations operating at a constant temporal scale.

To this end, we begin by studying the temporal redundancy in the learned feature representations. In particular, we analyze how the learned feature embeddings per speech frame are differentiated through the Conformer model depth. We randomly sample 100 audio signals from LibriSpeech's dev-other dataset, and process them through the Conformer blocks, recording their per-block activations. We then measure the average cosine similarity between two neighboring embedding vectors. The results are plotted as the solid lines in Fig. 3. We observe that the embeddings for the speech frames directly next to each other have an average similarity of 95% at the top layer, and even those 4 speech frames away from each other have a similarity of more than 80%. This reveals that there is an increasing temporal redundancy as inputs are processed through the Conformer blocks deeper in the network. We hypothesize that this redundancy in feature embedding vectors causes unnecessary computational overhead and the sequence length can be reduced deeper in the network without loss in accuracy.

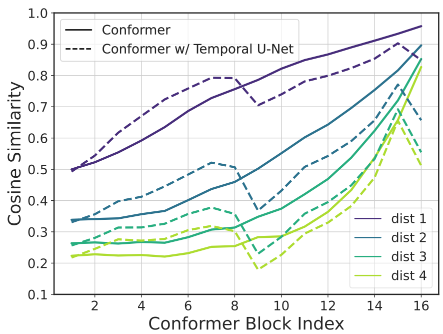

Figure 3: Cosine similarity between two embedding vectors of neighboring speech frames with varying adjacency distances across the Conformer blocks. The temporal dimension is downsampled after the 7th block and upsampled before the 16th block in the Temporal U-Net structure.

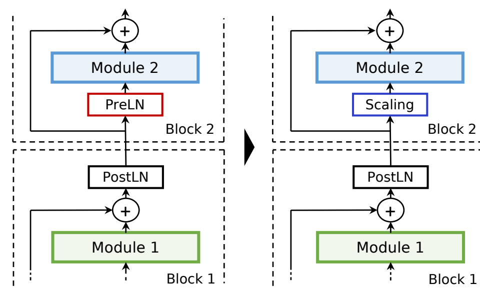

Figure 4: (Left) Back-to-back preLN and postLN at the boundary of the blocks. (Right) The preLN can be replaced with the adaptive scaling that readjusts the magnitude of the activation that goes into the subsequent module.

As our first macro-architecture improvement step, we change the Conformer model to incorporate subsampling of the embedding vectors after it has been processed by the early blocks of the model. In particular, we keep the sample rate to be 40ms up to the 7th block, and afterwards we subsample to a rate of 80ms per input sequence by using a pooling layer. For the pooling layer we use a depthwise separable convolution with stride 2 and kernel size 3 to merge the redundancies across neighboring embeddings. This decreases the attention complexity by $4\times$ and also reduces the redundancies of the features. This temporal downsampling shares similarities with computer vision models which often downsample the input image spatially to save compute and develop hierarchical level features [11, 18, 27, 45].

However, the temporal downsampling alone leads to divergence and an unstable training behaviour (§ 4.3). One possible reason for this is the lack of enough resolution for the decoder after subsampling the rate to 80ms. The decoder maps an embedding for each speech frame into a single label, e.g., character, and therefore requires sufficient resolution for successful decoding of the full sequence. Inspired from successful architectures for dense prediction in computer vision such as U-Net [42], we incorporate the Temporal U-Net structure to recover the resolution at the end of the network through an upsampling layer as shown in Fig. 2. This upsampling block takes the embedding vectors processed by the 40ms and 80ms sampling rates, and produces an embedding with a rate of 40ms by adding them together via a skip connection. To the best of our knowledge, the closest work to our Temporal U-Net is the approach proposed in [40], in which the U-Net structure is incorporated to a fully-convolutional model to downsample sleep signals.

This change not only reduces the total FLOPs by 20% compared to Conformer[1], but also improves the test-other WER by 0.62% from 7.90% to 7.28% (Tab. 1, 2nd row). Furthermore, analyzing the cosine similarity shows that the Temporal U-Net architecture prevents the neighboring embeddings

[1]The total FLOPs is for the entire model. If we just study the attention block, the Temporal U-Net structure reduces the FLOPs by $2.31\times$ and $2.53\times$ FLOPs reduction for processing 30s and 60s audio signals as compared to Conformer-CTC-M baseline, respectively.

from becoming too similar to each others at the later blocks, in particular at the final block directly connected to the decoder, as shown in Fig. 3 as the dashed lines.

3.1.2 Transformer-Style Block

The Conformer block consists of a sequence of feed-forward (‘F’), multi-head attention (MHA, ‘M’), convolution (‘C’), and another feed-forward module (‘F’). We denote this as the FMCF structure. Note that the convolutional kernel sizes in ASR models are rather large, e.g., 31 in Conformer, which makes its behaviour similar to attention in mixing global information. This is stark contrast to convolutional kernels in computer vision, that often have small 3×3 kernels and hence benefit greatly from attention's global processing. As such, placing the convolution and MHA module with a similar functionality back-to-back (i.e., the MC substructure) does not seem prudent. Hence, we consider an MF/CF structure, which is motivated by considering the convolution module as a local MHA module. Furthermore, we drop the Macaron structure [31], as MHA modules followed by feed-forward modules have been more widely adopted in the literature [8, 10, 41, 48]. In a nutshell, we simplify the architecture to be similar to the standard Transformer network and denote the blocks MF and CF substructures, as shown in Fig. 2. This modification further improves the test-other WER improvement by 0.16% from 7.28% to 7.12% and marginally improves the test-clean WER without affecting the FLOPs (Tab. 1, 3rd row).

3.2 Micro-Architecture Design

So far we have designed the macro structure of Squeezeformer by incorporating seminal architecture principles from computer vision and natural language processing into Conformer. In this subsection, we now focus on optimizing the micro structure of the individual modules. We show that we can further simplify the module architectures while improving both efficiency and performance.

3.2.1 Unified Activations

Conformer model uses Swish activation for most of the blocks. However, it switches to a Gated Linear Unit (GLU) for its convolution module. Such a heterogeneous design seems over-complicated and unnecessary. From a practical standpoint, multiple activations complicates hardware deployment, as an efficient implementation requires dedicated logic design, look up tables, or custom approximations [23, 36, 52]. To address this, we propose to replace the GLU activation with Swish, unifying the choice of activation function throughout the entire model. We keep the expansion rate for the convolution modules. As shown in the 4th row of Tab. 1, this change does not entail noticeable changes in WER and FLOPs but only simplifies the architecture.

3.2.2 Simplified Layer Normalizations

Continuing our micro-improvements, we note that the Conformer model incorporates redundant Layer Normalizations (LayerNorm), as shown in Fig. 4 (left). This is because the Conformer model contains both a Post-LayerNorm (PostLN) that applies LayerNorm in between the residual blocks, as well as Pre-LayerNorm (PreLN) which applies LayerNorm inside the residual connection. While it is hypothesized that preLN stabilizes training and postLN benefits performance [50], these two modules used together lead to redundant back-to-back operations. Aside from the architectural redundancy, LayerNorm can be computationally expensive [23, 52] due to its global reduction operations.

However, we found that straightforwardly removing the preLN or postLN leads to training instability and convergence failure (§ 4.3). Investigating the cause of failure, we observe that a typical trained Conformer model has orders of magnitude difference in the norm of the learnable scale variables of the back-to-back preLN and postLN. In particular, we found that the preLN would scale down the input signal by a large value, giving more weight to the skip connection. Therefore, it is important to use a scale layer when replacing the PreLN component to allow the network to control this weight. This idea is also on par with several training stabilization strategies in other domains. For instance, NF-Net [4] proposed adaptive (i.e., learnable) scaling before and after the residual blocks to stabilize training without normalization. Furthermore, DeepNet [50] also recently proposed to add non-trainable rule-based scaling to the skip connections, instead of the residual blocks, to stabilize PreLN in Transformers.

Table 2: Detailed architecture configurations for Conformer-CTC (baseline) and Squeezeformer. For comparison, we include the number of parameters and FLOPs for a 30s input in the last two columns.

Model	# Layers	Dimension	# Heads	Params (M)	GFLOPs
Conformer-CTC-S	16	144	4	8.7	26.2
Squeezeformer-XS	16	144	4	9.0	15.8
Squeezeformer-S	18	196	4	18.6	26.3
Conformer-CTC-M	16	256	4	27.4	71.7
Squeezeformer-SM	16	256	4	28.2	42.7
Squeezeformer-M	20	324	4	55.6	72.0
Conformer-CTC-L	18	512	8	121.5	280.6
Squeezeformer-ML	18	512	8	125.1	169.2
Squeezeformer-L	22	640	8	236.3	277.9

Inspired by these computer vision advancements, we propose to replace preLN with learnable scaling layer that scales and shifts the activations, $\mathrm{Scaling}(x) = \alpha x + \beta$, with learnable scale and bias vectors α and β of the feature dimension. For homogeneity of architectural design, we then replace the preLN throughout all the modules with the postLN-then-scaling as illustrated in Fig. 2 (Right) and make the entire model preLN-only. Note that the learned scaling p can be merged into the weights of the subsequent linear layer as the architecture illustrated in Fig. 2 (Right) and hence have zero inference cost. With the adaptive scaling our model further improves the test-other WER by 0.20% from 7.09% to 6.89% (Tab. 1, 5th row).

3.2.3 Depthwise Separable Subsampling

We now shift our focus from the Conformer blocks to the subsampling block. While it is easy to overlook this single module at the beginning of the architecture, we note that it accounts for a significant portion of the overall FLOPs count, up to 28% for Conformer-CTC-M with a 30-second input. This is because the subsampling layer uses two vanilla convolution operations each of which has a stride 2. To reduce the overhead of this layer, we replace the second convolution operation with a depthwise separable convolution while keeping the kernel size and stride the same. We leave the first convolution operation as is since it is equivalent to a depthwise convolution with the input dimension 1. This saves an additional 22% of the baseline FLOPs without a test-other WER drop and even a 0.06 improvement in test-clean WER (Tab. 1, 6th row). An important point to note here is that generally depthWise separable convolutions are hard to efficiently map to hardware accelerators, in part due its low arithmetic intensity. However, given the large FLOPs reduction, we consistently observe an overall reduction in the total model latency of about 30% as reported in Tab. 1, as compared to the baseline Conformer models.

We name our final model with all these improvements as Squeezeformer-SM. Compared to Conformer-CTC-M, our initial baseline, Squeezeformer-SM improves WER by 1.01% from 7.90% to 6.89% with 40% less FLOPs. Given the smaller FLOPs of Squeezeformer-SM, we also scale up the model to a similar FLOP cost as Conformer-CTC-M. In particular, we scale both depth and width of the model together following the practice in [9]. Scaling up the model achieves additional test-other WER gain of 0.39% from 6.89% to 6.50% (Tab. 1, 7th row), and we name this architecture Squeezeformer-M.

4 Results

4.1 Experiment Setup

Models. Following the procedure described in § 3, we construct Squeezeformer variants with different size and FLOPs: we apply the macro and micro-architecture changes in § 3.1 and § 3.2, respectively, to construct Squeezeformer-XS, SM, and ML from Conformer-S, M, and L retaining the model size. Afterwards, we construct Squeezeformer-S, M, and L by scaling up each model to match the FLOPs of the corresponding Conformer. The detailed architecture configurations are described in Tab. 2.

Table 3: WER (%) comparison on LibriSpeech dev and test datasets for Squeezeformer and other state-of-the-art ASR models including Conformer-CTC, QuartzNet [24], CitriNet [33], Transformer [28], and Efficient-Conformer [5] with and without the grouped attention (G.Att). For Conformer-CTC, the numbers are based on our own reproduction to the best performance as possible (marked *) due to the absence of public training recipes or codes. We report the numbers from NVIDIA's official public checkpoints [35] for QuartzNet and Citrinet, and the numbers reported in the paper for Efficient-Conformer. For comparison, we include the number of parameters, FLOPs, and latency on an NVIDIA's Tesla a100 GPU for a 30s input in the last three columns.

Model	dev-clean	dev-other	test-clean	test-other	Params (M)	GFLOPs	Latency (ms)
Conformer-CTC-S* [14]	4.21	10.54	4.06	10.58	8.7	26.2	1.63
QuartzNet 5x5 [24]	5.39	15.69	-	-	6.7	20.2	-
Citrinet 256 [33]	4.2	10.7	4.4	10.7	10.3	16.8	-
Squeezeformer-XS	**3.63**	**9.30**	**3.74**	**9.09**	9.0	15.8	1.31
Conformer-CTC-M* [14]	2.94	7.80	3.20	7.90	27.4	71.7	2.16
QuartzNet 5x10 [24]	4.14	12.33	-	-	12.8	38.5	-
QuartzNet 5x15 [24]	3.98	11.58	3.90	11.28	18.9	55.7	-
Citrinet 512 [33]	3.7	8.9	3.7	8.9	37.0	63.1	-
Eff. Conformer w/o G.Att [5]	-	-	3.57	8.99	13.2	26.0	-
Eff. Conformer w/ G.Att [5]	-	-	3.58	8.88	13.2	32.5	-
Squeezeformer-S	2.80	7.49	3.08	7.47	18.6	26.3	1.66
Squeezeformer-SM	**2.71**	**6.98**	**2.79**	**6.89**	28.2	42.7	1.79
Conformer-CTC-L* [14]	2.61	6.45	2.80	6.55	121.5	280.6	5.01
Citrinet 1024 [33]	3.7	8.3	3.6	7.9	143.1	246.3	-
Squeezeformer-M	2.43	6.51	**2.56**	6.50	55.6	72.0	2.32
Squeezeformer-ML	**2.34**	**6.08**	2.61	**6.05**	125.1	169.2	3.73
Transformer [28]	2.6	7.0	2.7	6.8	255.2	621.1	-
Squeezeformer-L	**2.27**	**5.77**	**2.47**	**5.97**	236.3	277.9	4.82

While there are multiple options available for the decoder such as RNN-Transducer (RNN-T) [12] and Connectionist Temporal Classification (CTC) [13], we use a CTC decoder whose non-autoregressive decoding method benefits training and inference latency [33]. However, the main focus of this work is the model architecture design of the encoder, which is completely orthogonal to the decoder type.

Another subtlety when evaluating models is the use of external language models (LM). In many prior work [16, 22, 32, 37, 51, 53], decoders are often augmented with external LMs such as pre-trained 4-gram or Transformer, which boosts the final WER by re-scoring the outputs in a more lexically accurate manner. However, we compare the results *without* external LMs to fairly compare the true representation power of the model architectures alone - external LMs can be incorporated as an orthogonal optimization afterwards.

Training Details. Because the training recipes and codes for Conformer have not been open-sourced, we train it to reproduce the best performance numbers as possible. We train both Conformer-CTC and Squeezeformer on the Librispeech-960hr [38] for 500 epochs on Google's cloud TPUs v3 with batch size 1024 for the small and medium variants and 2048 for the large variants. We use AdamW [30] optimizer with weight decay 5e-4 for all models. More details for the training and evaluation setup are given in § A.2 and § A.3.

4.2 Main Results

In Tab. 3 we compare the WER of Squeezeformer with Conformer-CTC and other state-of-the-art CTC-based ASR models including QuartzNet [24], CitriNet [33], Transformer [28], and Efficient Conformer [5] on the clean and other datasets. For simplicity, we denote WER as test-clean/test-other without % throughout the section.

Squeezeformer vs. Conformer. Our smallest model Squeezeformer-XS outperforms Conformer-CTC-S by 0.32/1.49 (3.74/9.09 vs. 4.06/10.58) with 1.66× FLOPs reduction. Compared with Conformer-CTC-M, Squeezeformer-S achieves 0.12/0.43 WER improvement (3.08/7.47 vs. 3.20/7.90) with 1.47× smaller size and 2.73× less FLOPs, and Squeezeformer-SM further improves

WER by 0.41/1.01 (2.79/6.89 vs. 3.20/7.90) with a comparable size and 1.70× less FLOPs. Compared with Conformer-CTC-L, Squeezeformer-M shows 0.24/0.05 WER improvement (2.56/6.50 vs. 2.80/6.55) with significant size and FLOPs reductions of 2.18× and 3.90×, respectively, and Squeezeformer-ML shows 0.19/0.50 WER improvement (2.61/6.05 vs. 2.80/6.55) with a similar size and 1.66× less FLOPs. Finally, our largest model Squeezeformer-L improves WER by 0.33/0.58 upon Conformer-CTC-L with the same FLOPs count, achieving the state-of-the-art result of 2.47/5.97.

Squeezeformer vs. Other ASR Models. As can be seen in Tab. 3, our model consistently outperforms QuartzNet, CitriNet, and Transformer with comparable or smaller model sizes and FLOPs counts. A notable result is a comparison against Efficient-Conformer: our model outperforms the efficiently-designed Efficient Conformer by a large margin of 0.49/1.52 (2.79/3.08 vs. 3.57/8.99) with the same FLOPs count. The overall results are summarized as a plot in Fig. 1 (Right) where Squeezeformer consistently outperforms other models across all FLOPs regimes.

4.3 Ablation Studies

In this section, we provide additional ablation studies for the design choices made for individual architecture components using Squeezeformer-M as the base model. Unless specified, we use the same hyperparameter settings with the main experiment.

Temporal U-Net. In the 2nd row of Tab. A.1, the model clearly under performs by 0.35/0.87 without the skip connection from the downsampling layer to the upsampling layer. This shows that the high-resolution information collected in the early layers is critical for successful decoding. The 3rd row in Tab. A.1 shows that our model completely fails to converge without the upsampling layer due to training stability, even with several different peak learning rates of {0.5, 1.0, 1.5}e-3.

LayerNorm. In the 4th line of Tab. A.1, we show that WER drops significantly by 3.17/7.49 when we apply the PostLN-only scheme without the learned scaling layer. Another alternative design choice is to apply the PreLN-only scheme without the learned scaling, which also results in a noticeable WER degradation of 0.59/1.76 as shown in the 5th line of Tab. A.1. In both cases, the model fails to converge, so we report the best WER before divergence. The results suggest that the learned scaling layer plays a key role for training stabilization and better WER.

Convolution Module. When ablating the GLU activation in the convolution modules, another possible design choice is to drop it without replacing with the Swish activation. This, however, results in 0.10/0.22 worse WER as shown in the last line of Tab. A.1.

5 Conclusions

In this work, we performed a series of systematic ablation studies on the macro and micro architecture of the Conformer architecture and proposed a novel hybrid attention-convolution architecture that is simpler, and consistently achieves better performance than other models for a wide range of computational budgets. The key novel components of Squeezeformer's macro-architecture is the incorporation of the Temporal U-Net structure which downsamples audio signals in the second half of the network to reduce the temporal redundancy between adjacent features and save compute, as well reorder the block order to MFCF be more similar to the standard Transformer-style MF/CF block structure which simplifies the architecture and improves performance. Furthermore, the micro-architecture of Squeezeformer simplifies the activations throughout the model and removes redundant LayerNorms with the Scaled Post-LN, which is more efficient and leads to better accuracy. We also drastically reduce the subsampling cost at the beginning of the model by incorporating a depth-wise separable convolution. We perform extensive testing of the proposed architecture and find that Squeezeformer scales very well across different model sizes and FLOP regimes, surpassing prior model architectures when trained under the same settings. Our code along with the checkpoints for all of the trained models is open sourced and available online [1].

References

[1] Anonymous. https://github.com/neurips2022-2985/squeezeformer.

[2] Alexei Baevski, Wei-Ning Hsu, Qiantong Xu, Arun Babu, Jiatao Gu, and Michael Auli. Data2vec: A general framework for self-supervised learning in speech, vision and language. *arXiv preprint arXiv:2202.03555*, 2022.

[3] Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli. wav2vec 2.0: A framework for self-supervised learning of speech representations. *Advances in Neural Information Processing Systems*, 33:12449–12460, 2020.

[4] Andy Brock, Soham De, Samuel L Smith, and Karen Simonyan. High-performance large-scale image recognition without normalization. In *International Conference on Machine Learning*, pages 1059–1071. PMLR, 2021.

[5] Maxime Burchi and Valentin Vielzeuf. Efficient conformer: Progressive downsampling and grouped attention for automatic speech recognition. *arXiv preprint arXiv:2109.01163*, 2021.

[6] Xuankai Chang, Aswin Shanmugam Subramanian, Pengcheng Guo, Shinji Watanabe, Yuya Fujita, and Motoi Omachi. End-to-end asr with adaptive span self-attention. In *INTERSPEECH*, pages 3595–3599, 2020.

[7] Sanyuan Chen, Chengyi Wang, Zhengyang Chen, Yu Wu, Shujie Liu, Zhuo Chen, Jinyu Li, Naoyuki Kanda, Takuya Yoshioka, Xiong Xiao, et al. Wavlm: Large-scale self-supervised pre-training for full stack speech processing. *arXiv preprint arXiv:2110.13900*, 2021.

[8] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[9] Piotr Dollár, Mannat Singh, and Ross Girshick. Fast and accurate model scaling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 924–932, 2021.

[10] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[11] Haoqi Fan, Bo Xiong, Karttikeya Mangalam, Yanghao Li, Zhicheng Yan, Jitendra Malik, and Christoph Feichtenhofer. Multiscale vision transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6824–6835, 2021.

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

[13] Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd international conference on Machine learning*, pages 369–376, 2006.

[14] 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. *arXiv preprint arXiv:2005.08100*, 2020.

[15] Pengcheng Guo, Florian Boyer, Xuankai Chang, Tomoki Hayashi, Yosuke Higuchi, Hirofumi Inaguma, Naoyuki Kamo, Chenda Li, Daniel Garcia-Romero, Jiatong Shi, et al. Recent developments on espnet toolkit boosted by conformer. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5874–5878. IEEE, 2021.

[16] Kyu J Han, Ramon Prieto, and Tao Ma. State-of-the-art speech recognition using multi-stream self-attention with dilated 1d convolutions. In *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 54–61. IEEE, 2019.

[17] Wei Han, Zhengdong Zhang, Yu Zhang, Jiahui Yu, Chung-Cheng Chiu, James Qin, Anmol Gulati, Ruoming Pang, and Yonghui Wu. ContextNet: Improving convolutional neural networks for automatic speech recognition with global context. *arXiv preprint arXiv:2005.03191*, 2020.

[18] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[19] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.

[20] Wei-Ning Hsu, Benjamin Bolte, Yao-Hung Hubert Tsai, Kushal Lakhotia, Ruslan Salakhutdinov, and Abdelrahman Mohamed. Hubert: Self-supervised speech representation learning by masked prediction of hidden units. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:3451–3460, 2021.

[21] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7132–7141, 2018.

[22] Shigeki Karita, Nanxin Chen, Tomoki Hayashi, Takaaki Hori, Hirofumi Inaguma, Ziyan Jiang, Masao Someki, Nelson Enrique Yalta Soplin, Ryuichi Yamamoto, Xiaofei Wang, et al. A comparative study on transformer vs rnn in speech applications. In *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 449–456. IEEE, 2019.

[23] Sehoon Kim, Amir Gholami, Zhewei Yao, Michael W Mahoney, and Kurt Keutzer. I-bert: Integer-only bert quantization. In *International conference on machine learning*, pages 5506–5518. PMLR, 2021.

[24] Samuel Kriman, Stanislav Beliaev, Boris Ginsburg, Jocelyn Huang, Oleksii Kuchaiev, Vitaly Lavrukhin, Ryan Leary, Jason Li, and Yang Zhang. QuartzNet: Deep automatic speech recognition with 1d time-channel separable convolutions. In *ICASSP*, pages 6124–6128. IEEE, 2020.

[25] Taku Kudo and John Richardson. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. *arXiv preprint arXiv:1808.06226*, 2018.

[26] Jason Li, Vitaly Lavrukhin, Boris Ginsburg, Ryan Leary, Oleksii Kuchaiev, Jonathan M Cohen, Huyen Nguyen, and Ravi Teja Gadde. Jasper: An end-to-end convolutional neural acoustic model. *arXiv preprint arXiv:1904.03288*, 2019.

[27] Yanghao Li, Chao-Yuan Wu, Haoqi Fan, Karttikeya Mangalam, Bo Xiong, Jitendra Malik, and Christoph Feichtenhofer. Improved multiscale vision transformers for classification and detection. *arXiv preprint arXiv:2112.01526*, 2021.

[28] Tatiana Likhomanenko, Qiantong Xu, Vineel Pratap, Paden Tomasello, Jacob Kahn, Gilad Avidov, Ronan Collobert, and Gabriel Synnaeve. Rethinking evaluation in asr: Are our models robust enough? *arXiv preprint arXiv:2010.11745*, 2020.

[29] Chunxi Liu, Frank Zhang, Duc Le, Suyoun Kim, Yatharth Saraf, and Geoffrey Zweig. Improving rnn transducer based asr with auxiliary tasks. In *2021 IEEE Spoken Language Technology Workshop (SLT)*, pages 172–179. IEEE, 2021.

[30] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

[31] Yiping Lu, Zhuohan Li, Di He, Zhiqing Sun, Bin Dong, Tao Qin, Liwei Wang, and Tie-Yan Liu. Understanding and improving transformer from a multi-particle dynamic system point of view. *arXiv preprint arXiv:1906.02762*, 2019.

[32] Christoph Lüscher, Eugen Beck, Kazuki Irie, Markus Kitza, Wilfried Michel, Albert Zeyer, Ralf Schlüter, and Hermann Ney. Rwth asr systems for librispeech: Hybrid vs attention–w/o data augmentation. *arXiv preprint arXiv:1905.03072*, 2019.

[33] Somshubra Majumdar, Jagadeesh Balam, Oleksii Hrinchuk, Vitaly Lavrukhin, Vahid Noroozi, and Boris Ginsburg. Citrinet: Closing the gap between non-autoregressive and autoregressive end-to-end models for automatic speech recognition. *arXiv preprint arXiv:2104.01721*, 2021.

[34] Edwin G Ng, Chung-Cheng Chiu, Yu Zhang, and William Chan. Pushing the limits of non-autoregressive speech recognition. *arXiv preprint arXiv:2104.03416*, 2021.

[35] NVDIA Nemo. https://github.com/nvidia/nemo.

[36] NVDLA Primer. http://nvdla.org/primer.html, 2021.

[37] Jing Pan, Joshua Shapiro, Jeremy Wohlwend, Kyu J Han, Tao Lei, and Tao Ma. Asapp-asr: Multistream cnn and self-attentive sru for sota speech recognition. *arXiv preprint arXiv:2005.10469*, 2020.

[38] Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. 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, 2015.

[39] 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. *arXiv preprint arXiv:1904.08779*, 2019.

[40] Mathias Perslev, Michael Jensen, Sune Darkner, Poul Jørgen Jennum, and Christian Igel. U-time: A fully convolutional network for time series segmentation applied to sleep staging. *Advances in Neural Information Processing Systems*, 32, 2019.

[41] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training. 2018.

[42] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[43] Kyuhong Shim, Jungwook Choi, and Wonyong Sung. Understanding the role of self attention for efficient speech recognition. In *International Conference on Learning Representations*, 2021.

[44] Laurent Sifre and Stéphane Mallat. Rigid-motion scattering for texture classification. *arXiv preprint arXiv:1403.1687*, 2014.

[45] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[46] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *International Conference on Machine Learning*, pages 10347–10357. PMLR, 2021.

[47] Vincent Vanhoucke. Learning visual representations at scale. 2014.

[48] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[49] Chengyi Wang, Yu Wu, Yao Qian, Kenichi Kumatani, Shujie Liu, Furu Wei, Michael Zeng, and Xuedong Huang. Unispeech: Unified speech representation learning with labeled and unlabeled data. In *International Conference on Machine Learning*, pages 10937–10947. PMLR, 2021.

[50] Hongyu Wang, Shuming Ma, Li Dong, Shaohan Huang, Dongdong Zhang, and Furu Wei. Deepnet: Scaling transformers to 1,000 layers. *arXiv preprint arXiv:2203.00555*, 2022.

[51] Yongqiang Wang, Abdelrahman Mohamed, Due Le, Chunxi Liu, Alex Xiao, Jay Mahadeokar, Hongzhao Huang, Andros Tjandra, Xiaohui Zhang, Frank Zhang, et al. Transformer-based acoustic modeling for hybrid speech recognition. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6874–6878. IEEE, 2020.

[52] Joonsang Yu, Junki Park, Seongmin Park, Minsoo Kim, Sihwa Lee, Dong Hyun Lee, and Jungwook Choi. Nn-lut: Neural approximation of non-linear operations for efficient transformer inference. *arXiv preprint arXiv:2112.02191*, 2021.

[53] Frank Zhang, Yongqiang Wang, Xiaohui Zhang, Chunxi Liu, Yatharth Saraf, and Geoffrey Zweig. Faster, simpler and more accurate hybrid asr systems using wordpieces. *arXiv preprint arXiv:2005.09150*, 2020.

[54] 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 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7829–7833. IEEE, 2020.

[55] Shucong Zhang, Erfan Loweimi, Peter Bell, and Steve Renals. Stochastic attention head removal: A simple and effective method for improving transformer based asr models. *arXiv preprint arXiv:2011.04004*, 2020.

[56] Shucong Zhang, Erfan Loweimi, Peter Bell, and Steve Renals. On the usefulness of self-attention for automatic speech recognition with transformers. In *2021 IEEE Spoken Language Technology Workshop (SLT)*, pages 89–96. IEEE, 2021.

[57] Yu Zhang, James Qin, Daniel S Park, Wei Han, Chung-Cheng Chiu, Ruoming Pang, Quoc V Le, and Yonghui Wu. Pushing the limits of semi-supervised learning for automatic speech recognition. *arXiv preprint arXiv:2010.10504*, 2020.

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [No]
   (c) Did you discuss any potential negative societal impacts of your work? [No]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] As anonymous github URL
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [No] Due to the prohibitive amount of compute requirement for a single run of training, we do not report the numbers from multiple runs. However, our improvement over the baseline is significantly large to be affected by the random seeds.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] We added citations for the dataset and the baseline models
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]