Lookahead When It Matters: Adaptive Non-causal Transformers for Streaming Neural Transducers

Grant P. Strimel [*1] Yi Xie [*1] Brian King [*1] Martin Radfar [1] Ariya Rastrow [1] Athanasios Mouchtaris [1]

Abstract

Streaming speech recognition architectures are employed for low-latency, real-time applications. Such architectures are often characterized by their causality. Causal architectures emit tokens at each frame, relying only on current and past signal, while non-causal models are exposed to a window of future frames at each step to increase predictive accuracy. This dichotomy amounts to a trade-off for real-time Automatic Speech Recognition (ASR) system design: profit from the low-latency benefit of strictly-causal architectures while accepting predictive performance limitations, or realize the modeling benefits of future-context models accompanied by their higher latency penalty. In this work, we relax the constraints of this choice and present the Adaptive Non-Causal Attention Transducer (ANCAT). Our architecture is non-causal in the traditional sense, but executes in a low-latency, streaming manner by dynamically choosing when to rely on future context and to what degree within the audio stream. The resulting mechanism, when coupled with our novel regularization algorithms, delivers comparable accuracy to non-causal configurations while improving significantly upon latency, closing the gap with their causal counterparts. We showcase our design experimentally by reporting comparative ASR task results with measures of accuracy and latency on both publicly accessible and production-scale, voice-assistant datasets.

1. Introduction

Modern speech recognition applications have leveraged the benefits afforded by fully-neural architectures to drive enhanced user experiences. For example, these architectures allow popular virtual assistants such as Amazon Alexa, Google Assistant, and Apple Siri to rely on robust, far-field speech recognition as their primary medium of real-time interaction. Similarly, offline services such as recorded call transcription and video caption generation apply these architectures as well. The dominant techniques like Neural-Transducers, Listen-Attend-Spell, and Transformers are all principally sequence-to-sequence models consisting of an audio encoder mapping acoustic input to high-level representations for autoregressive decoding (Graves, 2012; Chan et al., 2016; Mohamed et al., 2019; Chorowski et al., 2015; Han et al., 2020). And while each neural ASR architecture has its individual design elements, generally speaking, each is categorized or implemented as a causal or non-causal model.

Causal ASR models are streamable designs which emit predictions for each frame as they arrive from the audio signal, without access to future frames (He & al., 2019; Yu et al., 2021a; Radfar et al., 2022). These models are referred to as causal in the sense that each frame prediction is only dependent on its left (past) context. Causal configurations are particularly attractive for real-time processing applications where fast response times are essential for the user experience, such as virtual assistants or live stream caption generation. Non-causal models, on the other hand, have access to information from future frames. Non-causal models can be either streaming with a bounded number of lookahead frames or full-context with access to the entirety of the audio signal, before beginning to decode (Zhang et al., 2020; Moritz et al., 2020; Yeh et al., 2019; Tripathi et al., 2020). Naturally, these models can deliver significantly improved accuracy over their causal counterparts by leveraging future information to disambiguate predictions holistically. The accuracy gains of non-causal processing typically are accompanied by a steep price of higher latency in real-time applications however, which can hinder these models for production deployment.

In this paper, we introduce the Adaptive Non-Causal Attention Transducer (ANCAT), a neural ASR architecture which has the accuracy improvements witnessed by non-causal models, while yielding a latency that is more comparable

[*]Equal contribution [1]Amazon Alexa, USA. Correspondence to: Grant P. Strimel <gsstrime@amazon.com>, Yi Xie <yixiey@amazon.com>.

Proceedings of the 40^{th} International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023. Copyright 2023 by the author(s).

to streaming causal architectures. ANCAT achieves this improvement by providing flexibility to the model to rely on variable future context at each frame. However, it is trained to be selective about how and when it does so, taking into account both the accuracy and latency implications to poll for future context only where necessary to make accurate predictions. The resulting behavior is that the model's latency costs for adaptively ingesting lookahead context, when aggregated over the entire frame sequence, is comparatively minimal. Our model is trained fully end-to-end, jointly learning how and where to apply non-causal context in tandem with its traditional token prediction objectives.

After summarizing motivating related work in Section 2, the remainder of the paper is organized according to our contributions: Section 3 introduces the ANCAT design and key architectural concepts, Section 4 derives novel loss functions developed for training our architecture and regularizing future context in connection with different notions of latency commonly applied in the literature, and Section 5 presents empirical results which justify our design elements and showcase the modeling capabilities of ANCAT for streaming ASR tasks on both open-source and industrial data.

2. Related Work

Causal, Non-causal, and Streamable ASR. Bridging the accuracy benefits of non-causality with the low-latency deployability of causal models has been the focus of many prior studies. Several works (Audhkhasi et al., 2021; Moritz et al., 2021; Yu et al., 2021b) find that distillation techniques leveraging non-causal right context benefit the training of fully causal models. For instance, "dual-mode" ASR, where a single end-to-end model is jointly trained with full-context mode using shared weights, improves causal ASR accuracy (Yu et al., 2021b).

Other approaches enable more accurate streaming by permitting ingestion of a finite amount of frame-wise lookahead context at the cost of a latency penalty. For example, using a fixed right-context window of lookahead frames at each layer is common (Zhang et al., 2020). Chunking approaches are also effective for non-causality (Tsunoo et al., 2019; Dong et al., 2019; Shi et al., 2021; Chen et al., 2021). With chunking, the stream is broken down into fixed-sized groupings of non-overlapping, adjacent frames. Within a chunk, all frames have access to one another and possibly frames from prior chunks. Similar to dual-mode training, (Swietojanski et al., 2022) also extends in-place distillation to chunking of variable sizes, training a single model but allowing different chunk sizes to be configured at inference to match hardware specifications. Scout-networks (Wang et al., 2020), meanwhile, use word boundaries to define the position and sizes of chunks and use a separate network trained on forced alignment data to predict boundaries at inference time.

Another set of approaches unify causal and non-causal context into one system by training streaming and full-context encoders and stacking them. (Narayanan et al., 2021) and (Li et al., 2021) propose "cascaded" audio encoders where a causal first-pass encoder is followed by a stacked non-causal encoder operating on the first encoder's outputs. These modules are jointly learned to produce accuracy improvements, but accept the latency impact from the full-context second-pass encoder.

Adaptive Neural Compute. Adaptive compute, also referred to commonly as variable or dynamic compute, is a technique that adjusts the amount of neural computation a model executes during inference as a function of each individual input. The motivating intuition of the approach is that since each instance has different characteristics, the corresponding amount of computation expended should reflect this variety, conditioning for more resources and operations only where necessary. Researchers have explored these ideas extensively across machine learning areas, including NLP (Graves, 2016; Jernite et al., 2017; Dehghani et al., 2019; Elbayad et al., 2020), vision (Bolukbasi et al., 2017; Figurnov et al., 2017), speech (Macoskey et al., 2021a;b; Xie et al., 2022; Peng et al., 2023) and recommendation systems (Song et al., 2019). We refer readers to (Han et al., 2021) for a comprehensive survey. While (Sukhbaatar et al., 2020; Chang et al., 2020) also propose learning static attention span adjustment, our mechanism will vary the span across inputs and frames adaptively for streaming. Additionally, all adaptive techniques effectively tie their compute to a particular cost metric, such as floating point operations. For our application, we will show how to link our dynamic compute mechanism to key latency measures for real-time speech recognition.

Latency Measures. There are intricacies to assessing the latency of a streaming ASR architecture, and as a result, numerous metrics have been proposed to encapsulate its various facets. User-perceived latency (UPL), endpointer latency, first token emission delay, algorithmic latency, mean alignment delay, and partial recognition latency are all among those measures which are considered in the literature (Shangguan et al., 2021; Sainath & al., 2020; Inaguma et al., 2020; Yu et al., 2021a). While each definition attempts to capture a different latency driver, for models using non-causal context, it is natural to use algorithmic latency, and therefore, we also adopt this measure to inform the design of our adaptive non-causal streaming architecture. *Algorithmic latency* reports the time required processing the input to produce the output of an audio frame – frame length combined with the amount of lookahead frames used (Shi et al., 2021). Since our algorithm is nondeterministic, we adjust to reporting a statistical version of the metric. We also consider a *compute-induced UPL* metric in our design, detailed in Section 4.3, which jointly accounts for processing speed and additional work scheduled by our non-causal mechanism to derive their combined impact on UPL.

3. Adaptive Non-causal Design

In this work we focus on the Neural Transducer architecture consisting classically of three modules: an encoder network, a prediction network, and a joint network. The encoder network takes a sequence of T feature frame vectors $(x_1, \dots, x_T)$ extracted from the audio signal and maps it to a corresponding sequence of high-level acoustic representations. The prediction network operates autoregressively over a sequence of U labels $(y_1, \dots, y_U)$. The joint network combines the outputs from the encoder and prediction networks to model the likelihood for the next label emitted.

The transducer architecture is a popular choice for real-time applications because the encoder network can be trained for streaming, allowing the joint network to emit its prediction on the i-th frame relying on a bounded number of future frames; however, full-context encoders can be applied to yield additional accuracy for non-streaming scenarios. Our design utilizes Transformer-based encoders which are constructed from L stacked blocks (a.k.a. layers, terms which we use interchangeably), each of which accepts the output from the previous block and produces hidden vector h_i^ℓ for frame i and layer ℓ. Transformer-based blocks consist of various compositions of sublayers such as feed-forwards, layer-norms, and even convolutions in the case of Conformer (Gulati et al., 2020), but all contain the multi-headed self-attention (MHSA) sublayer.

3.1. Compute DAGs and Attention Masking

Transformer-based attention architectures define a natural compute graph of input dependencies for each output of each layer. Under this directed acyclic graph (DAG) representation, each vertex v_i^ℓ (or node) represents the compute on frame i of layer ℓ, and a directed edge $\left(v_j^{\ell-1}, v_i^\ell\right)$ between vertices of adjacent layers indicates the reliance on the output of $v_j^{\ell-1}$ for computing v_i^ℓ, namely that $h_j^{\ell-1}$ (or a mapping thereof) is attended over when computing h_i^ℓ.

Our ANCAT architecture is designed to dynamically and strategically fill in the edges of this compute DAG, balancing both accuracy and latency considerations. To accomplish this, we train the traditional encoder weights while introducing jointly trainable schedulers $\mathcal{S}^\ell$ into the architecture, one for each layer ℓ. Each scheduler $\mathcal{S}^\ell$ accepts the prior layer's result $h_i^{\ell-1}$ and determines the future, non-causal inputs to attend over to compute h_i^ℓ. Hence, $\mathcal{S}^\ell(h_i^{\ell-1})$ predicts the non-causal edges from vertices $v_j^{\ell-1}$ to vertex v_i^ℓ for positions $j > i$.

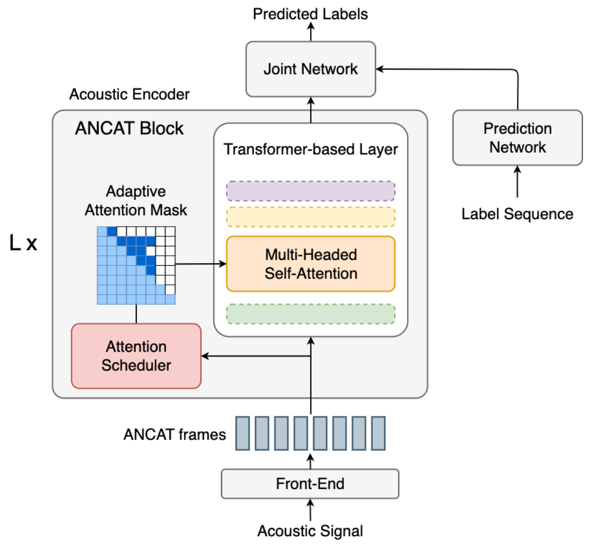


Figure 1. Adaptive Non-Causal Attention Transducer (ANCAT). The architecture is a neural transducer with an acoustic encoder of L stacked Transformer-based blocks where each layer is augmented with an attention scheduler. Each scheduler learns to fill in the right context attention connections and passes the resulting mask to the multi-headed self-attention.

During training, we leverage a series of attention masks $M^\ell \in [0,1]^{T \times T}$ to represent these edges, where $M_{i,j}^\ell = 1$ indicates the full presence of the $(v_j^{\ell-1}, v_i^\ell)$ edge and 0 indicates its absence:

$$M_{i,j}^\ell = \begin{cases} 1 & j \leq i \\ \mathcal{S}^\ell(h_i^{\ell-1})_j & \text{otherwise} \end{cases} \qquad 1 \leq i, j \leq T$$

The mask is applied for computing attention scores across each of the attention heads in the MHSA sublayer. Namely, for a scaled dot product value matrix P computed from query matrix X_{query} and key matrix X_{key} for a particular head

$$P = \frac{X_{\text{query}} X_{\text{key}}^\top}{\sqrt{d}},$$

the masked attention scores can be computed as

$$A_{i,j} = \frac{\exp(P_{i,j}) M_{i,j}^\ell}{\sum_t^T \exp(P_{i,t}) M_{i,t}^\ell} \qquad 1 \leq i, j \leq T$$

for each attention head of layer ℓ or using method of (Xie et al., 2022).

3.2. Learned Schedulers

The role of the scheduler is to fill in the forward (right of the main diagonal) values of its corresponding attention mask. To simplify the learning process, instead of predicting each connection individually, we view each scheduler $\mathcal{S}$ as estimating the length of the non-causal attention span to apply at its layer (i.e., the number of future frames to consider) on a particular input. We employ a smooth, differentiable, reverse "S"-shaped function decreasing from 1 to 0 with an adjustable parameter τ to control the sharpness of the curve. The schedulers learn where to shift the center o of this curve over a span of K maximum lookahead frames. Specifically, $\mathcal{S}^\ell$ is computed as

$$o_i^\ell = \sigma\left(\mathrm{FFN}^\ell(h_i^{\ell-1})\right)(K+\epsilon)$$
$$\mathcal{S}^\ell(h_i^\ell)_j = \begin{cases} 1-\sigma\left(\left(j-i-o_i^\ell\right)/\tau\right) & j \le K \\ 0 & \text{otherwise,} \end{cases}$$

where σ is the standard sigmoid function and FFN^ℓ is a learnable feed-forward network consisting of two linear transforms with a non-linear activation in between, with the second transform projecting to a single scalar. The small constant ϵ is used for enabling the Kth lookahead frame to potentially have a near-full connection while maintaining numeric stability during training.

Note that the design permits "soft edges" during the learning process with mask values between 0 and 1. However, τ can be annealed towards 0 to gradually morph all soft edges into definitive binary ones. Our loss function design will leverage this property as well.

4. Regularizing Future Context

Complementing the design of the ANCAT architecture, we craft several loss functions which regularize the decisions of the schedulers to explicitly account for different notions of latency for streaming systems.

4.1. Naive Regularization

To balance both the primary training objective of the neural transducer with the incurred impact of the schedulers' ingestion of non-causal frames, one can modify the training loss function to the form

$$\mathcal{L} = \mathcal{L}_{\text{transducer}} + \lambda\mathcal{L}_{\text{sched}}.$$

A first attempt would be to simply regularize over all lookahead connections, with the intuition that attending over the fewest possible future frames across layers will yield lower latency

$$\mathcal{L}_{\text{L1}} = \frac{1}{T}\sum_{\ell}\sum_{j>i} M_{i,j}^\ell$$

This L1-type regularization will serve as a baseline for our experiments. However, indiscriminately regularizing in this manner does not account for a connection's non-local effects on latency. For example, a forward edge present in a lower

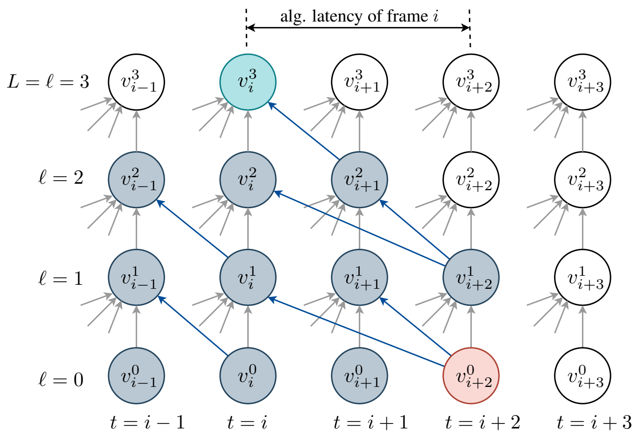


Figure 2. Compute graph example of ANCAT for three layers. Non-causal edges (blue) are provided by the learned schedulers. Dependency nodes (gray) of v_i^L (cyan) determine the algorithmic latency. The difference $\delta(i)$ between the final dependency frame (red) and v_i^L determines the algorithmic latency for frame i.

level can dramatically impact the prediction delay over many frames because of the propagating effects layer over layer. Furthermore, the simple regularization above is not directly tied to an explicit latency measure.

4.2. Algorithmic Latency

Here, we formulate how to regularize with respect to the notion of algorithmic latency. Recall that traditionally, algorithmic latency would refer to the minimal theoretic amount of time the model needs to wait before it can emit a symbol at a particular frame, resulting from the number of non-causal future frames relied upon $\delta(i)$ for prediction at each timestep i along with the frame length (which is a constant).[1] This delay is a deterministic value for standard architectures. In contrast, with our ANCAT architecture, the algorithmic latency can vary from frame to frame, and we therefore adopt a time-averaged mean definition:

$$\text{algorithmic latency} = \frac{1}{T}\sum_i(\text{frame length})\delta(i)$$

We derive an algorithm below to directly integrate the algorithmic latency stemming from the schedulers' decisions and aggregate the result into a regularization loss we apply during training.

To do so, we first define the notion of a dependency function. We define $D_{i,j}^\ell \in [0,1]$ to represent the (fractional) dependency for computing h_i^ℓ on the input node v_j^0. In other words, $D_{i,j}^\ell$ represents if v_j^0 is connected to v_i^ℓ by some

[1]It is common to include the "current" i-th frame in this calculation; however, for clarity of presentation under our setup, we omit it. But since this singular frame amounts to an additive constant, one can easily translate any result to account for it.

pathway through the compute DAG. The dependency values can be fractional because the edges will be fractional during training.

For the ANCAT architecture, we propose the following memoized dynamic programming formulation using the attention masks to recursively compute the dependency matrices at each layer

$$D_{i,j}^{\ell} = \begin{cases} M_{i,j}^{\ell} & \ell = 1 \\ \max\limits_{t} \; M_{i,t}^{\ell} \cdot D_{t,j}^{\ell-1} & \ell > 1 \end{cases}$$

This algorithm can be viewed as computing the maximum fractional strength of a pathway between an input frame and a compute node. Interestingly, this algorithm becomes a special case reduction of a classic shortest path in a graph problem using edge weight from $v_j^{\ell-1}$ and v_i^{ℓ} as $-\ln M_{i,j}^{\ell}$ and with the final $D_{i,j}^{\ell}$ values extracted as $\exp\left\{-\text{distance}\left(v_j^0, v_i^{\ell}\right)\right\}$.

Importantly, for our application, the algorithm produces dependency matrices for each layer ℓ that has $D_{i,j}^{\ell}$ as monotonically decreasing in j. The following argument by induction proves this fact: The base case is straight forward because $D_{i,j}^1 = M_{i,j}^1$ and by scheduler function monotonicity. Now let $D_{i,j}^{\ell} = M_{i,t}^{\ell} \cdot D_{t,j}^{\ell-1}$ for some t. Since $D_{t,j-1}^{\ell-1} \geq D_{t,j}^{\ell-1}$ by induction hypothesis, there exists at least one t', namely $t' = t$, over which a maximum is taken such that $D_{i,j-1}^{\ell} \geq M_{i,t'}^{\ell} \cdot D_{t',j-1}^{\ell-1} \geq M_{i,t}^{\ell} \cdot D_{t,j}^{\ell-1} = D_{i,j}^{\ell}$.

As a result of this monotonicity, we treat $D_{i,j}^{\ell}$ as an approximate, inverse cumulative distribution function over j where $\hat{P}(\text{last dependency frame of } v_i^{\ell} \leq t) = 1 - D_{i,t}^{\ell}$. Therefore, the corresponding density function $F_{i,j}^{\ell} = D_{i,j-1}^{\ell} - D_{i,j}^{\ell}$ provides a natural weighting for the position of the final lookahead frame required to compute node v_i^{ℓ}.

Observe that indeed $\sum_j F_{i,j}^{\ell} = 1$ for all i, ℓ and that the distribution turns into a strictly one-hot vector coding which indicates the exact position of the last frame dependency of v_i^{ℓ} as τ is annealed to produce binary connections from the schedulers.

We now can use this algorithm to directly regularize mean (fractional) algorithmic latency over all frames of the utterance

$$\mathcal{L}_{\text{Alg.Lat.}} = \frac{1}{T} \sum_i \hat{\delta}(i) = \frac{1}{T} \sum_{j>i} (j - i) \, F_{i,j}^{L}$$

using $F_{i,:}^{L}$ to indicate (weight) the last frame dependencies of the top layer L at each frame.

4.3. Compute-Induced UPL

While algorithmic latency as a metric provides perspective on the minimum delay a streaming ASR model is to expect, it operates under the assumption that all compute is instantaneous, and therefore, only serves as a lower bound of the model's response UPL. To more realistically model and regularize with respect to UPL, we propose a loss which also accounts for processing speed.

We denote μ as the effective processor speed in terms of compute nodes (encoder layers) which can be processed each second and ρ as the feature frame rate in frames per second. We use these constants and the algorithm presented in Section 4.2 to compute at which timestep each node becomes available to establish an accounting of the compute backlog. The final amount of work remaining (nodes left to compute) in the backlog on the final frame will then be used to regularize the UPL.

Letting q_i to be the number of nodes which have their input dependencies met on frame i, we have

$$q_i = \sum_{\ell} \sum_{t<i} F_{t,i}^{\ell}$$

which is the amount of new work to be added to the backlog at i. Defining b_i to be the buffered node backlog (lag accumulated at the i-th timestep), using a method similar to that of (Macoskey et al., 2021b), we express the compute-induced UPL loss as the response delay in seconds $\mathcal{L}_{\text{UPL}} = b_T/\mu$ with b_i derived recursively as

$$b_i = \begin{cases} 0 & i = 0 \\ \max \; \{b_{i-1} + q_i - \mu/\rho, 0\} & i > 0 \end{cases}$$

where μ/ρ is the compute throughput in terms of how many nodes per timestep can be burned down in the backlog. Figure 3 illustrates how this UPL computation is carried out.

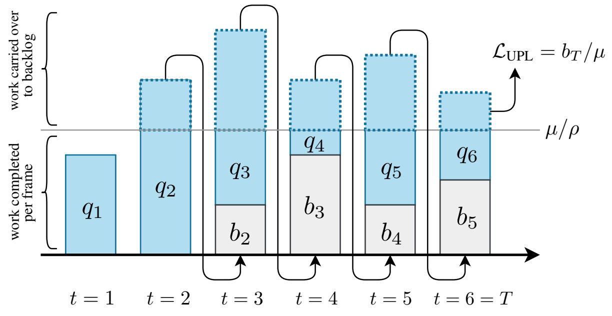


Figure 3. Example of the UPL algorithm in action. Work which cannot be completed on the current frame under a specified throughput is carried over to the subsequent frame. The remaining work at the end of the utterance defines the UPL.

5. Experimental Results

We organize our results in this section to demonstrate ANCAT's combined accuracy and algorithmic latency improvements on publicly available ASR data, supplemented with a

Table 1. Results on LibriSpeech test "clean" for Conformer and T-T encoder backbones. The models are trained with different streaming settings for maximum lookahead frames K on each layer, while the full context model is also reported. *Layerwise* and *Chunked* represent standard lookahead attention structures and chunked-aware attention methods of prior literature. ANCAT models trained with L1 regularization and our novel loss are shown as ANCAT-*L1* and ANCAT-*Alg.Lat.*, respectively. ANCAT-*Alg.Lat.* improves WER by significant margins over nearest competitors with comparable mean algorithmic latency and likewise for 50th and 90th percentiles.

	Conformer													
K (# lookahead)	0	2				5				10				Full
Model	*Causal* ●	*Layerwise* ●	*Chunked* ●	ANCAT-*L1* ●	ANCAT-*Alg.Lat.* ●	*Layerwise* ●	*Chunked* ●	ANCAT-*L1* ●	ANCAT-*Alg.Lat.* ●	*Layerwise* ●	*Chunked* ●	ANCAT-*L1* ●	ANCAT-*Alg.Lat.* ●	*Full Context* ●
WER (%)	6.66	4.58	6.27	5.68	**5.31**	4.23	5.68	5.43	**5.13**	4.12	5.15	5.13	**4.76**	4.09
Alg.Latency (ms)	0.00	2240	63.2	60.5	60.1	3240	235	235	202	3520	518	495	498	-
Alg.Latency@50 (ms)	0.00	2380	59.9	61.6	58.8	2940	236	223	237	2940	518	488	489	-
Alg.Latency@90 (ms)	0.00	2920	62.4	86.4	76.7	5680	241	310	274	6650	541	705	623	-
ℓ_1-norm (frames)	0.00	27.1	6.94	1.78	8.12	65.3	27.3	7.04	16.5	122	60.3	16.2	59.1	-
	Transformer													
WER (%)	6.90	4.88	6.56	5.92	**5.57**	4.63	5.74	5.67	**5.36**	4.42	5.65	5.27	**5.11**	4.55
Alg.latency (ms)	0.00	2240	63.2	61.2	60.5	3240	235	232	229	3520	518	512	496	-
Alg.Latency@50 (ms)	0.00	2380	59.9	60.4	55.8	2940	236	230	228	2940	518	508	484	-
Alg.Latency@90 (ms)	0.00	2920	62.4	88.6	80.2	5680	241	269	305	6650	541	701	613	-
ℓ_1-norm (frames)	0.00	27.1	6.94	1.89	7.93	65.3	27.3	5.47	21.5	122	60.3	18.5	36.3	-

summary of findings on industry data, model dynamics, and compute-induced UPL studies while referring the reader to corresponding appendices for their full results.

5.1. LibriSpeech Experimental Setup

We investigate our architectures using the LibriSpeech corpus (Panayotov et al., 2015) comprised of 960 hours of training data collected from read audio books. Our evaluations report on the associated "clean" test data. Audio clips are preprocessed with a 64-dimensional log-filterbank energy feature extractor, and these feature vectors are stacked with a stride size of 2 and downsampled by 3 before being processed by a small convolution front-end to produce 120ms frames as input for the transformer-based blocks.

We conduct our experiments on two popular transformer-based architectures: Conformer (Gulati et al., 2020) and Transformer-Transducer (T-T) (Zhang et al., 2020). For both Conformer and T-T, we use encoders consisting of 14 stacked blocks, one-layer 640-unit LSTM prediction networks, and a 512-unit feed-forward joint network. For our ANCAT variants, we augment each block with learnable schedulers of 64-dimension hidden units. The detailed model configurations are listed in Appendix G.

For LibriSpeech, all the models are trained for 150 epochs of 1k steps with the Transformer-based encoder backbone and schedulers trained end-to-end jointly using the Adam optimizer (Kingma & Lei Ba, 2015) using the same hyperparameters settings specified by (Gulati et al., 2020). During training, we also apply SpecAugment (Park et al., 2019) with mask parameter ($F = 27$) and 20 time masks with maximum time mask ratio ($p_S = 0.04$), where the maximum-size of the time mask is set to p_S times the length of the utterance. We use a vocabulary of 4097 word pieces and the standard RNN-T beam search decoding with a width of 16. Further specifications on model configurations and training hyperparameters are shared in Appendix G.

5.2. Baselines

We establish baseline models for Conformer and T-T models using causal and full-context attention. We also build baselines for non-causal streaming mechanisms proposed in prior studies. These include streaming attention which ingests K lookahead frames at each layer (Zhang et al., 2020), denoted here as *Layerwise*, and chunking transformer attention (Shi et al., 2021; Chen et al., 2021), denoted as *Chunked*, which is a popular and efficient approach for comparison. Additionally, we present ANCAT models with basic L1-type regularization over the amount of future frames as described in Section 4.1, marked as ANCAT-*L1*.

5.3. Algorithmic Latency Results

Our main empirical result showcases the significant improvements on each operating point of the latency-accuracy trade-off curve that ANCAT produces when trained with our novel regularization loss, and thereby, shifting the Pareto frontier over existing methods. Namely, for each fixed accuracy target, our ANCAT model produces significant decreases in algorithmic latency over the nearest baseline, and conversely, for a desired algorithmic latency, ANCAT also yields significant improvements in accuracy. We establish this result by training the baseline and ANCAT models under various hyperparameter settings for the maximum per-layer lookahead span K (i.e., 2, 5, and 10). We report Word Error Rate (WER) (%) and the statistical algorithmic latency (ms) of test utterances. We also report on latency with median algorithmic latency and 90th percentile metrics, denoted as Alg.Latency@50 and Alg.Latency@90. Included is the ℓ_1-norm of the accessible future frames in attention maps to characterize the additional total amount of forward attention calculations (additional computation over causal). The aggregation means are taken across all utterances in the test set.

Results for LibriSpeech are arranged in Table 1. While the algorithmic latency values for the non-adaptive models are static for a given K, the WER and latency trade-off can be tuned by setting different penalty degrees on the regularizing term λ for ANCAT models. To make clear comparisons of WER in Table 1, we choose λ values so that under each setting of K the Alg.Latency approximately matches that of the most efficient comparable baselines. As can be seen, training with our proposed ANCAT using our novel algorithmic latency loss as the regularizer, ANCAT-*Alg.Lat.*, consistently yields lower WER over the approaches for equivalent latency operating points. This holds for both Conformer and T-T encoders and all settings K, seeing an average of 11% (and upwards of 18% for the lower latency scenarios) relative WER improvement over non-adaptive models. Observe also, that compared with naive L1 regularization, for a given algorithmic latency budget, our ANCAT-*Alg.Lat.* models allow for a greater total amount of future frame attention based on ℓ_1-norm measures. So while ANCAT-*Alg.Lat.* uses overall higher compute, it is more strategic in how it expends it to deliver better WER for matching latency.

To further highlight the improvements in the trade-off between WER and algorithmic latency, we record Conformer ANCAT model performance at multiple levels of training regularization in Figure 4. The plot shows that ANCAT-*Alg.Lat.* consistently outperforms all other architectures at each operating point, and therefore fully defines the Pareto frontier of efficient solutions, providing the most optimal trade-offs of those models under consideration. Furthermore, the plot emphasizes that non-causal adaptivity alone *does not* provide this improvement since ANCAT-*L1* closely matches static *Chunked* performance; rather, ANCAT adaptivity paired with the proper choice of our novel regularization method is critical to the success of the approach under latency considerations.

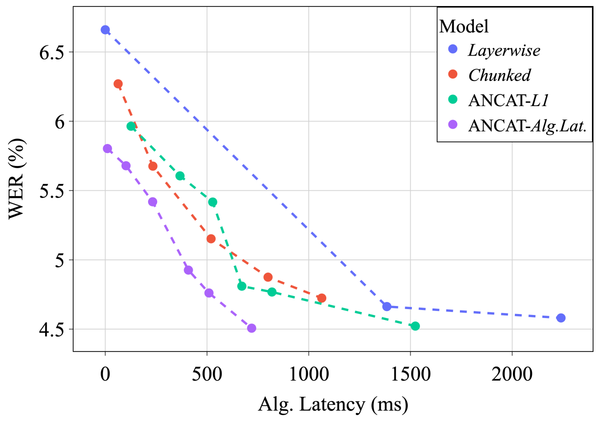


Figure 4. WER vs. algorithmic latency on LibriSpeech test "clean" data for four different types of Conformer models: *Layerwise* (K at 0, 1, and 2 frames of right context per layer), *Chunked* (K at 2, 5, 10, 15, and 20 frames), ANCAT-*L1* ($K = 10$) varying λ, and ANCAT-*Alg.Lat.* ($K = 10$) varying λ. ANCAT-*Alg.Lat.* provides a more optimal accuracy-latency trade-off over other models.

For further insight into the behavior of our proposed algorithmic latency loss and its impact on scheduling, we visualize the attention masks of an utterance from the test set in Figure 5 and compare against the static approaches. The attention masks are generated with the final training epoch where $\tau = 1e-4$. The darker areas indicate the absence of attention. We depict the maps from the 3rd block and the 13th block in the Conformer models. One can observe that ANCAT-*Alg.Lat.* learns to structure its attention in stepwise patterns which effectively operates like a chunking strategy of varying sizes and locations conditioned on the input. In contrast, ANCAT-*L1* attention patterns are jagged and emphasizes more localized decisions of the schedulers as opposed to the better coordinated decisions of ANCAT-*Alg.Lat.* accounting for the global impact on latency. We also show masks of an ANCAT model trained with higher latency penalization to demonstrate greater toggling-off of future attention connections, bordering being a fully causal model.

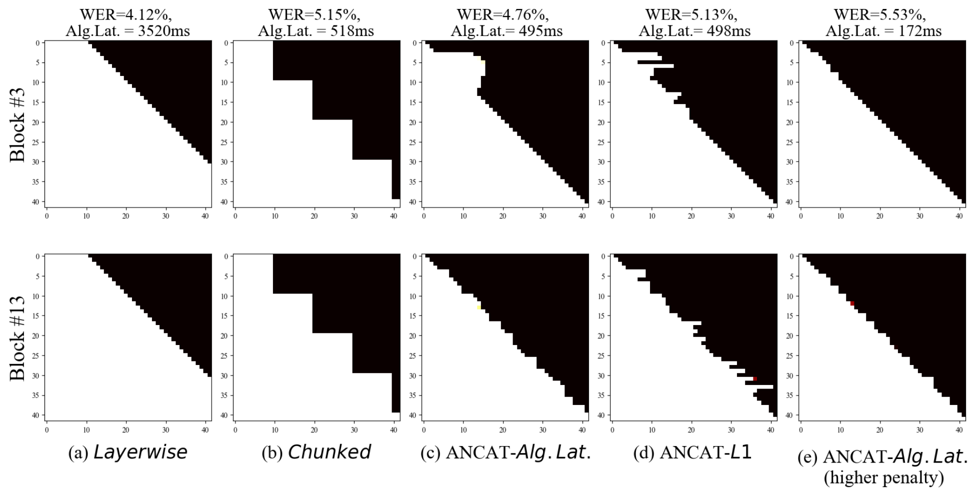


Figure 5. Attention mask visualization for the 3rd and 13th blocks of Conformer models. All the attention masks are generated with the same test example utterance. This utterance has a total number of 41 frames after downsampling in the front-end. The max number of lookahead frames is $K = 10$. The darker regions signal toggled-off attention. We select (a), (b), and (c) models to have comparable algorithmic latency performance, and (d) shows a model that is optimized with a higher latency penalty. Notice the deliberate, structured stepwise patterns ANCAT-*Alg.Lat.* emits (c,e) compared to the jagged schedules produce by L1 regularization (d).

5.4. Industrial Voice Assistant

We repeat our algorithmic latency experiments on a de-identified large, industrial voice assistant dataset for again Conformer and T-T to verify robustness of our modeling approach both across architectures and data compositions. We find similar results to that of LibriSpeech with ANCAT-*Alg.Lat.* consistently outperforming *Layerwise*, *Chunked*, and ANCAT-*L1* models at each operating point and improving upon WER by often greater than 5% relative. Appendix C details this analogous experimental setup and its results.

5.5. Compute-Induced UPL

In addition to applying our algorithmic latency loss as the regularization method, we also conduct experiments to regularize with respect to compute-induced UPL, both for LibriSpeech and industrial data. These results are presented in Appendix D and mirror our findings with algorithmic latency, with ANCAT regularized with the backlog latency method presented in Section 4.3 outperforming the baselines. We find that for all compute throughput settings on which we experimented, for a given UPL budget, ANCAT delivers superior accuracy, with typically ANCAT-*UPL* improving WER by over 8% relative on LibriSpeech and 7% on industrial data over the nearest baselines.

5.6. Additional Findings and Analysis

We present supplemental visualizations and observations of the approach in further appendices. Appendix A demonstrates that ANCAT promotes superior performance over baselines when comparing with other common latency metrics. Appendix B shows that ANCAT also performs well (10% WER relative improvement compared with baselines) in more challenging conditions using LibriSpeech "other" data and additive noise. We further show how the difficulty of an utterance correlates to the degree of lookahead employed. Appendix E highlights and expands upon additional attention plot examples while Appendix F depicts how the characteristics of ANCAT evolve over the course of training by temperature annealing. The figures show that our "soft" attention connections and latency measures smoothly converge throughout training with the binary edge runtime latency calculations.

6. Conclusion

In this work, we introduce an adaptive non-causal architecture for streaming speech recognition. The model learns to dynamically adjust the future context attention span for each individual frame of the audio stream, balancing both accuracy and latency considerations. We accompanied our architecture construction with novel regularizing loss functions which tie the frame-wise lookahead decisions of the model with key latency measures important for speech applications. The resulting mechanism provides a better Pareto frontier of trade-offs against baselines, in many cases with over 15% relative WER improvements for matching latency. Our experiments on public and large, production datasets and different architectures reinforce the robustness and applicability of our approach. We hope that future work will build on these contributions to propose adaptive non-causal approaches for other applications, measures of latency, and models of computation.

The authors would like to thank Gautam Tiwari for his expert guidance and recommendations.

References

Audhkhasi, K., Chen, T., Ramabhadran, B., and Moreno, P. J. Mixture model attention: Flexible streaming and non-streaming automatic speech recognition. *Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH)*, pp. 4096–4100, 2021.

Bolukbasi, T., Wang, J., Dekel, O., and Saligrama, V. Adaptive neural networks for efficient inference. *International Conference on Machine Learning (ICML)*, pp. 812–821, 2017.

Chan, W., Jaitly, N., Le, Q. V., and Vinyals, O. Listen, attend and spell. *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4960–4964, 2016.

Chang, X., Subramanian, A. S., Guo, P., Watanabe, S., Fujita, Y., and Omachi, M. End-to-end ASR with adaptive span self-attention. *Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH)*, pp. 3595–3599, 2020.

Chen, X., Wu, Y., Wang, Z., Liu, S., and Li, J. Developing real-time streaming transformer transducer for speech recognition on large-scale dataset. *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5904–5908, 2021.

Chorowski, J., Bahdanau, D., Serdyuk, D., Cho, K., and Bengio, Y. Attention-based models for speech recognition. *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 577–585, 2015.

Dehghani, M., Gouws, S., Vinyals, O., Uszkoreit, J., and Kaiser, Ł. Universal transformers. *International Conference on Learning Representations (ICLR)*, pp. 1–23, 2019.

Dong, L., Wang, F., and Xu, B. Self-attention aligner: A latency-control end-to-end model for ASR using self-attention network and chunk-hopping. *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5656–5660, 2019.

Elbayad, M., Gu, J., Grave, E., and Auli, M. Depth-adaptive transformer. *International Conference on Learning Representations (ICLR)*, 2020.

Figurnov, M., Collins, M. D., Zhu, Y., Zhang, L., Huang, J., Vetrov, D., and Salakhutdinov, R. Spatially adaptive computation time for residual networks. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1790–1799, 2017.

Graves, A. Sequence transduction with recurrent neural networks. *International Conference on Machine Learning (ICML)*, 2012.

Graves, A. Adaptive computation time for recurrent neural networks. *arXiv preprint arXiv:1603.08983*, 2016.

Gulati, A., Qin, J., Chiu, C. C., Parmar, N., Zhang, Y., Yu, J., Han, W., Wang, S., Zhang, Z., Wu, Y., and Pang, R. Conformer: convolution-augmented transformer for speech recognition. *Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH)*, pp. 5036–5040, 2020.

Han, W., Zhang, Z., Zhang, Y., Yu, J., Chiu, C. C., Qin, J., Gulati, A., Pang, R., and Wu, Y. ContextNet: Improving convolutional neural networks for automatic speech recognition with global context. *Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH)*, pp. 3610–3614, 2020.

Han, Y., Huang, G., Song, S., Yang, L., Wang, H., and Wang, Y. Dynamic neural networks: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44 (11):7436–7456, 2021.

He, Y. and al., E. Streaming end-to-end speech recognition for mobile devices. *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 6381–6385, 2019.

Inaguma, H., Gaur, Y., Lu, L., Li, J., and Gong, Y. Minimum latency training strategies for streaming sequence-to-sequence ASR. *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 6064–6068, 2020.

Jernite, Y., Grave, E., Joulin, A., and Mikolov, T. Variable computation in recurrent neural networks. *International Conference on Learning Representations (ICLR)*, pp. 1–12, 2017.

Kingma, D. P. and Lei Ba, J. Adam: A method for stochastic optimization. *International Conference on Learning Representations (ICLR)*, 2015.

Li, B., Gulati, A., Yu, J., Sainath, T. N., Chiu, C. C., Narayanan, A., Chang, S. Y., Pang, R., He, Y., Qin, J., Han, W., Liang, Q., Zhang, Y., Strohman, T., and Wu, Y. A better and faster end-to-end model for streaming ASR. *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5634–5638, 2021.

Macoskey, J., Strimel, G. P., and Rastrow, A. Bifocal neural asr : exploiting keyword spotting for inference optimization. *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5999–6003, 2021a.

Macoskey, J., Strimel, G. P., Su, J., and Rastrow, A. Amortized neural networks for low-latency speech recognition. *Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH)*, pp. 1868–1872, 2021b.

Mohamed, A., Okhonko, D., and Zettlemoyer, L. Transformers with convolutional context for ASR. *arXiv preprint arXiv:1904.11660*, 2019.

Moritz, N., Hori, T., and Roux, J. L. Streaming automatic speech recognition with the transformer model. *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 6074–6078, 2020.

Moritz, N., Hori, T., and Le Roux, J. Dual causal/non-causal self-attention for streaming end-to-end speech recognition. *Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH)*, pp. 4116–4120, 2021.

Narayanan, A., Sainath, T. N., Pang, R., Yu, J., Chiu, C. C., Prabhavalkar, R., Variani, E., and Strohman, T. Cascaded encoders for unifying streaming and non-streaming ASR. *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5629–5633, 2021.

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

Park, D. S., Chan, W., Zhang, Y., Chiu, C.-c., Zoph, B., Cubuk, E. D., and Le, Q. V. Specaugment: A simple data augmentation method for automatic speech recognition. *arXiv preprint arXiv:1904.08779*, 2019.

Peng, Y., Lee, J., and Watanabe, S. I3D: Transformer architectures with input-dependent dynamic depth for speech recognition. *arXiv preprint arXiv:2303.07624*, 2023.

Radfar, M., Barnwal, R., Swaminathan, R. V., Chang, F. J., Strimel, G. P., Susanj, N., and Mouchtaris, A. ConvRNN-T: Convolutional Augmented Recurrent Neural Network Transducers for Streaming Speech Recognition. *Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH)*, pp. 4431–4435, 2022.

Sainath, T. N. and al., E. A streaming on-device end-to-end model surpassing server-side conventional model quality and latency. *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 6059–6063, 2020.

Shangguan, Y., Prabhavalkar, R., Su, H., Mahadeokar, J., Shi, Y., Zhou, J., Wu, C., Le, D., Kalinli, O., Fuegen, C., and Seltzer, M. L. Dissecting user-perceived latency of on-device E2E speech recognition. *arXiv preprint arXiv:2104.02207*, 2021.

Shi, Y., Wang, Y., Wu, C., Yeh, C.-F., Chan, J., Zhang, F., Le, D., and Seltzer, M. Emformer: efficient nemory transformer based acoustic model for low latency streaming speech recognition. *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 6783–6787, 2021.

Song, W., Charlin, L., Xiao, Z., Zhang, M., Wang, Y., and Tang, J. Session-based social recommendation via dynamic graph attention networks. *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining (WSDM)*, pp. 555–563, 2019.

Sukhbaatar, S., Grave, E., Bojanowski, P., and Joulin, A. Adaptive attention span in transformers. *Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 331–335, 2020.

Swietojanski, P., Braun, S., Can, D., da Silva, T. F., Ghoshal, A., Hori, T., Hsiao, R., Mason, H., McDermott, E., Silovsky, H., Travadi, R., and Zhuang, X. Variable attention masking for configurable transformer transducer speech recognition. *arXiv preprint arXiv:2211.01438*, 2022.

Tripathi, A., Kim, J., Zhang, Q., Lu, H., and Sak, H. Transformer transducer: One model unifying streaming and non-streaming speech recognition. *arXiv preprint arXiv:2010.03192*, 2020.

Tsunoo, E., Kashiwagi, Y., Kumakura, T., and Watanabe, S. Towards online end-to-end transformer automatic speech recognition. *arXiv preprint arXiv:1910.11871*, 2019.

Wang, C., Wu, Y., Lu, L., Liu, S., Li, J., Ye, G., and Zhou, M. Low latency end-to-end streaming speech recognition with a scout network. *Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH)*, pp. 2112–2116, 2020.

Xie, Y., Macoskey, J., Radfar, M., Chang, F. J., King, B., Rastrow, A., Mouchtaris, A., and Strimel, G. P. Compute Cost Amortized Transformer for Streaming ASR. *Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH)*, pp. 3043–3047, 2022.

Yeh, C.-F., Mahadeokar, J., Kalgaonkar, K., Wang, Y., Le, D., Jain, M., Schubert, K., Fuegen, C., and Seltzer, M. L. Transformer-transducer: End-to-end speech recognition with self-attention. *arXiv preprint arXiv:1910.12977*, 2019.

Yu, J., Chiu, C.-C., Li, B., Chang, S.-y., Sainath, T. N., He, Y., Narayanan, A., Han, W., Gulati, A., Wu, Y., and Pang, R. FastEmit: Low-latency streaming ASR with sequence-level emission regularization. *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 6004–6008, 2021a.

Yu, J., Han, W., Gulati, A., Chiu, C.-C., Li, B., Sainath, T. N., Wu, Y., and Pang, R. Dual-mode ASR: Unify and improve streaming ASR with full-context modeling. *International Conference on Learning Representations (ICLR)*, 2021b.

Zhang, Q., Lu, H., Sak, H., Tripathi, A., Mcdermott, E., Koo, S., and Kumar, S. Transformer transducer: a streamable speech recognition model with transformer encoders and RNN-T loss. *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 7829–7833, 2020.

A. Emission Latency and Endpointer Latency

To conduct a more comprehensive investigation into the improved latency performance resulting from the ANCAT architecture, we carry out measurements on additional emission/transcript (EM) latency and endpointer (EP) latency in this section. EM.Latency refers to the mean difference between the frame in which a token is produced by decoding (accounting for future lookahead) and when the token is spoken given by alignment data. EP.Latency refers specifically to the difference between when an end-of-speech token is emitted and the final word is spoken (again, accounting for future lookahead). In Table 2, the numbers show comparable (or better) EM.Latency and EP.Latency to the best baseline, while the WER of our proposed ANCAT-*Alg.Lat.*is superior.

Table 2. Results on LibriSpeech test "clean" for Conformer. The models in this table are identical with the models in Table 1 in main body of the paper. We report the average emission/transcript (EM) latency and endpointer latency (EP) in this table.

	Conformer												
K (# lookahead)	0	2				5				10			
Model	*Causal* ●	*Layerwise* ●	*Chunked* ●	ANCAT-*L1* ●	ANCAT-*Alg.Lat.* ●	*Layerwise* ●	*Chunked* ●	ANCAT-*L1* ●	ANCAT-*Alg.Lat.* ●	*Layerwise* ●	*Chunked* ●	ANCAT-*L1* ●	ANCAT-*Alg.Lat.* ●
WER (%)	6.66	4.58	5.68	6.27	**5.31**	4.23	5.68	5.43	**5.13**	4.12	5.15	5.13	**4.76**
EM.Latency (ms)	235	2130	305	296	**290**	3125	395	315	**282**	3418	622	560	**551**
EP.Latency (ms)	74	1992	141	116	**116**	2992	252	172	**139**	3286	490	402	**408**

B. LibriSpeech "Other" and Noise Impact

The LibriSpeech corpus also provides the more difficult test "other" dataset comprised of 5.1 hours of speech signals chosen from more a challenging speaker cohort. Results of our ANCAT and baseline models are presented in Figure 6. Besides the naturally higher error rates for all models, we observe a very similar plot to that of test "clean" with ANCAT-*Alg.Lat.* providing the most optimal accuracy-latency trade-offs. Again, ANCAT-*Alg.Lat.* defines the Pareto frontier with respect to algorithmic latency and WER for those models considered while yielding upwards of 10% relative WER improvements for several operating points.

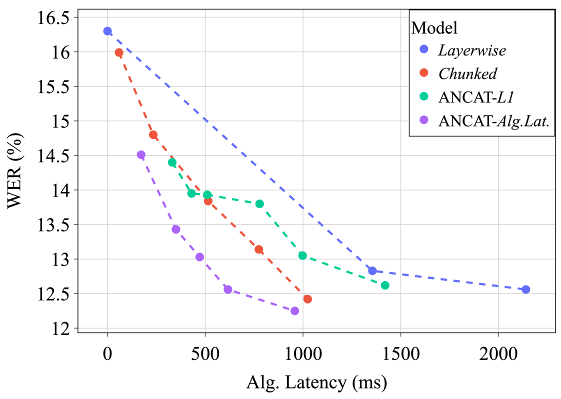


Figure 6. WER vs. algorithmic latency on LibriSpeech test "other" data for four different types of Conformer models: *Layerwise* (K at 0, 1, and 2 frames of right context per layer), *Chunked* (K at 2, 5, 10, 15, and 20 frames), ANCAT-*L1* ($K = 10$) varying λ, and ANCAT-*Alg.Lat.* ($K = 10$) varying λ. ANCAT-*Alg.Lat.* provides a more optimal accuracy-latency trade-off over other models.

One observes that while WER increases for test "other" data, the algorithmic latency also increases noticeably as well. This leads us to examine for a specified ANCAT model, the relationship between WER and algorithmic latency. Do naturally more difficult utterances (producing higher WER) correlate with inflated algorithmic latency driven by ANCAT schedulers? Figure 7 (left) answers this in the affirmative. The visualization is constructed by grouping all individual test "clean" and "other" utterance WERs into five percentiles. The WER of each of these groups is then compared with its mean latency to observe a positive association between the two quantities. This finding suggests that for more challenging utterances, ANCAT indeed will resort to using more lookahead attention. We also show a simple experiment in Figure 7 (right) which adds different degrees of noise to LibriSpeech test "clean" data and illustrates a smooth trend between latency and WER/noise. This finding supports the notion that under more challenging conditions, ANCAT-*Alg.Lat.* models will opt to use more lookahead attention to compensate for the added ambiguity.

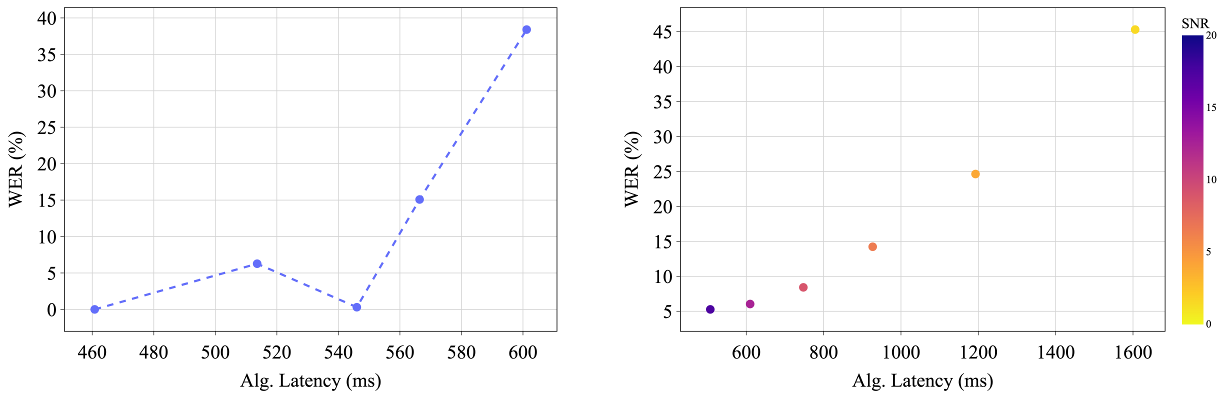


Figure 7. WER vs. algorithmic latency for single ANCAT-*Alg.Lat.* model. The left sub-figure is LibriSpeech test "clean" and "other" utterances are grouped into five percentiles based on individual WER with the aggregate WER within each group then plotted against its mean latency. The right sub-figure shows this same model performing on the test "clean" dataset with different levels of white noise added over the audio signals. There is a clear trend between the difficulty of the utterance and the amount of lookahead employed resulting in higher latency for more challenging examples.

C. Voice Assistant Data

In this section we discuss the experimental setup for Voice Assistant data and algorithmic latency results mirroring those of LibriSpeech. Further appendices will present additional findings and figures inclusive of both LibriSpeech and Voice Assistant experiments.

C.1. Experimental Setup

Our in-house datasets consist of de-identified, far-field, voice assistant utterances. The training dataset consists of approximately 150k hours of transcribed audio. We used model architectures identical to those we built for LibriSpeech, with just a couple of configuration differences better fitting for the voice assistant data. Specifically, we used a slightly larger word piece tokenization model of size 4k and the two layer convolutional front-end had stride sizes of 2 and 1 (resulting in 60 ms ANCAT frames) as opposed to 2 and 2 stride sizes for LibriSpeech. All our accuracy numbers for internal data we report as Word Error Rate Relative (WERR) with respect to the baseline fully causal model (below 7.5% WER absolute).

C.2. Algorithmic Latency

Plots for Conformer results pertaining to algorithmic latency are presented in the following figures. We couple figures with a final table in Appendix H for reference. Figure 8 visualizes how WER and algorithmic latency vary between the different types of models and different regularization factors. As done for Figure 4, we adjust the lookahead span for the static models *Layerwise* and *Chunked* while fixing K at 10 frames per layer for the ANCAT models and adjusting their regularization penalty. These results match closely with those of LibriSpeech with ANCAT-*Alg.Lat.* providing the most optimal accuracy-latency operating points. In many cases, ANCAT-*Alg.Lat.* outperforms its nearest competitor by 5% relative in WER for matching latency budgets. Again, one finds that ANCAT-*L1* performs comparable with *Chunked* and demonstrates that the novel latency regularization algorithm is vital to the adaptive approach's success over that of a more naive regularization scheme.

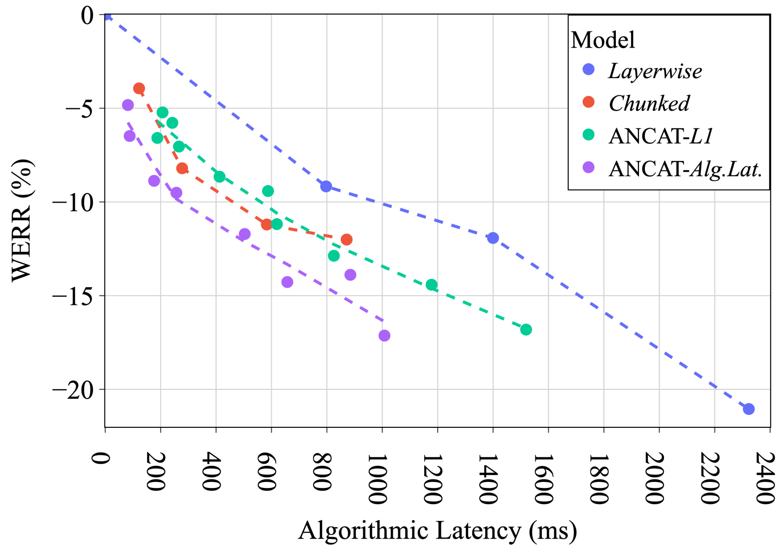


Figure 8. WERR vs. algorithmic latency on Voice Assistant data for four different types of Conformer models: 1) *Layerwise* (K at 0, 1, 2, and 10 frames of right context per layer), *Chunked* (K at 5, 10, 20, and 30 frames), ANCAT-*L1* ($K = 10$) varying regularization term λ, and ANCAT-*Alg.Lat.* ($K = 10$) varying λ.

To complement Figure 8 above and emphasise the tuneability of ANCAT, Figure 9 shows how WER and latency behave as a function of the regularization constant λ for Conformer ANCAT-*Alg.Lat.* model on Voice Assistant data. From this figure, one can see that the WERR and latency curves are relatively smooth over the weights, so it is easy to select a desired accuracy-latency operating point. Finally, note that ANCAT models with zero to high regularization are indeed close to the operating points expected: near zero latency for high regularization and significant accuracy improvements similar to maximal right context models for minimal regularization.

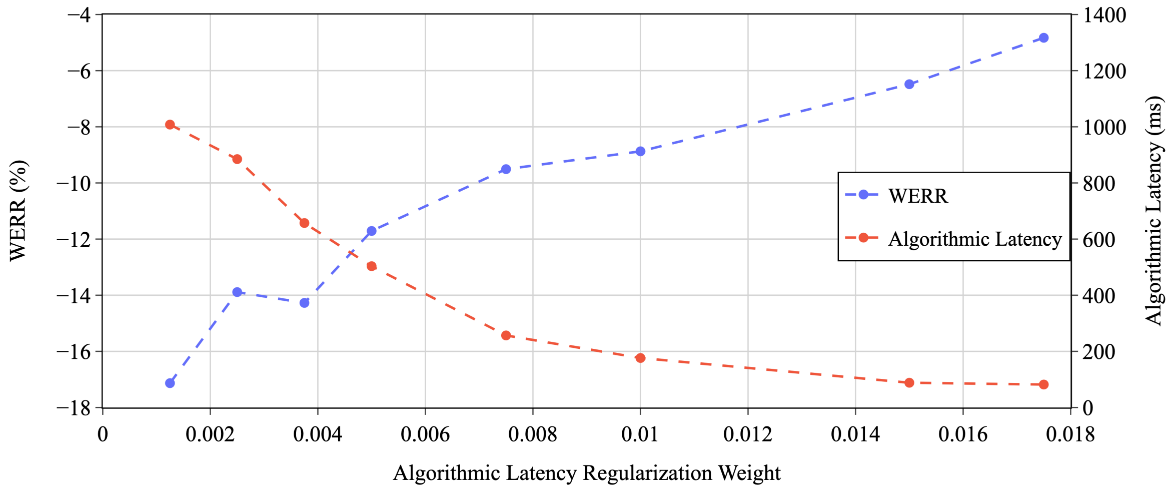


Figure 9. WERR and algorithmic latency vs. ANCAT regularization weight λ for a Conformer ANCAT-*Alg.Lat.* model on Voice Assistant data.

D. Compute-Induced UPL Results

Section 4.3 provides a novel regularization method to account for both the processing power of executing hardware along with the schedulers' decisions in order to reduce the approximate UPL an application might expect with an ANCAT model. Using this notion of latency, we experiment with LibriSpeech and Voice Assistant data to build ANCAT-*UPL* models with UPL regularization. For our 14-block model architecture and frame rate ρ, the experiments consider compute throughput values μ of 1.5, 2, and 3 ($\times 14\rho$). The values represent the number of nodes in the compute DAG which can be processed per second and intuitively are equivalent to the speed of 1.5, 2 and 3 forward passes through the model per frame (i.e. real-time multipliers).

In Figures 10 and 11, we see how WER and compute UPL vary between models with different regularization weights and compute throughput. We see that for a fixed throughput value, as the regularization is decreased, the UPL smoothly increases, analogous to our algorithmic latency results. Further we find that for both LibriSpeech and Voice Assistant, ANCAT-*UPL* provides the most efficient trade-off between accuracy and latency by a significant margin. ANCAT-*UPL* consistently outperforms the other architectures at each operating point, and fully defines the Pareto frontier of efficient solutions. It is typical for ANCAT-*UPL* to improve WER by over 8% relative on LibriSpeech and 7% on Voice Assistant of the nearest competitor for fixed UPL budgets.

These figures also show that providing ANCAT-*UPL* models with more compute power naturally leads to better trade-offs between UPL and WER. The higher throughput enables the models to apply lookahead attention to a greater degree without paying as high of a latency penalty since when the compute does become available (all DAG dependencies met), it executes quicker. Furthermore, under this definition of latency, as long as the backlog of compute remaining is minimal on the final frame of each utterance, models have the ability to attend to future context as much as they want as long as the work can be burned down before the end of the utterance. As the figures highlight, the more compute throughput the model is given, the better it can take advantage of this property to improve its WER.

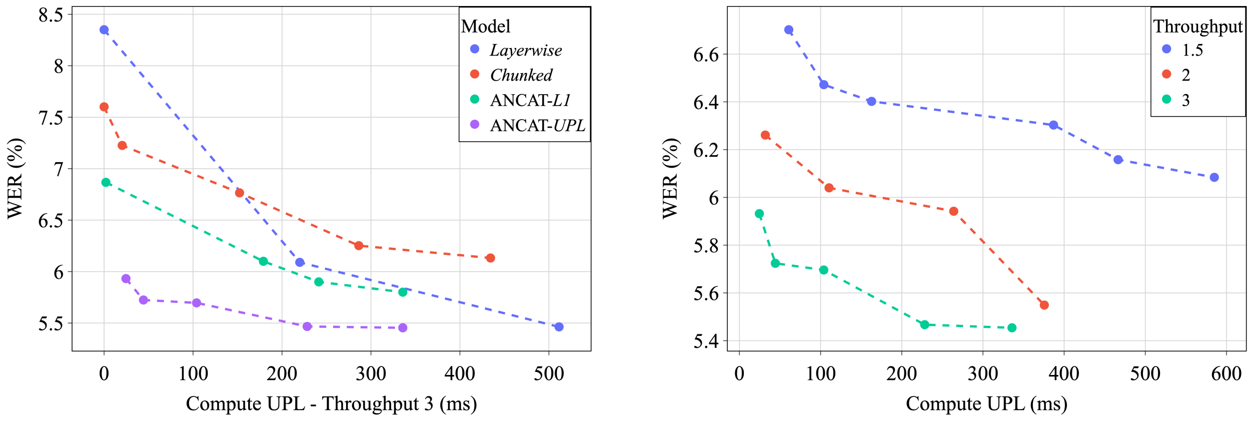
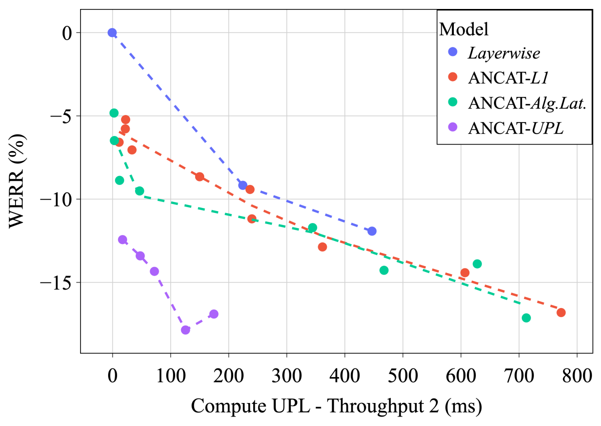


Figure 10. LibriSpeech test "clean": Compute UPL vs. WER. (Left) For a fixed throughput, UPL from the compute backlog comparing four different types of Conformer models: 1) *Layerwise* attention models (K at 0, 1, and 2 frames of right context per layer), 2) *Chunked* (K at 2, 5, 10, 15 and 20 frames of right context per layer), ANCAT-*L1*, and 4) ANCAT-*UPL* with ANCAT models having $K = 10$ and varying regularization term λ. With respect to perceived latency, ANCAT-*UPL* provides the superior accuracy-latency trade-off. (Right) ANCAT-*UPL* models trained with a fixed regularization factor but with different throughputs. Faster compute power gives the model more lookahead flexibility, resulting in better accuracy-latency operating points for ANCAT-*UPL* models. From our empirical observations, the UPL loss behaves sensitively time masking patterns and is prone to overfitting on small datasets. Thereby, for UPL results we only apply frequency masking of the SpecAugmentation which leads a slightly higher WER over our other LibriSpeech experiments.



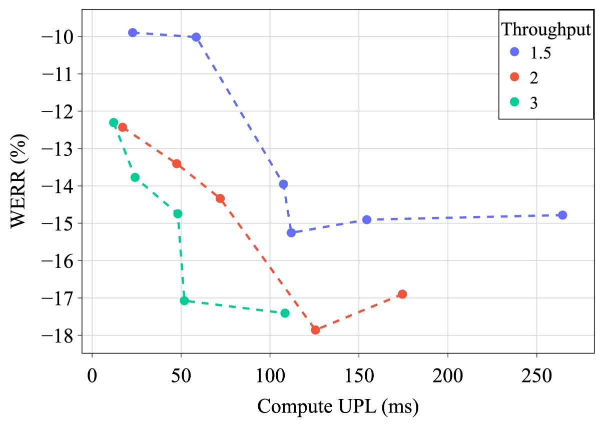
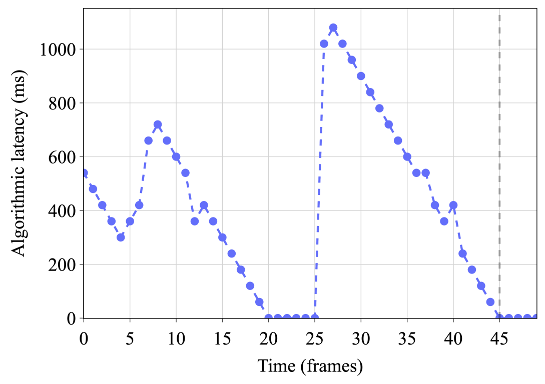


Figure 11. Voice Assistant: Compute UPL vs. WERR. (Left) For a fixed throughput, UPL from the compute backlog comparing four different types of Conformer models: 1) *Layerwise* attention models (K at 0, 1, and 2 frames of right context per layer), 2) ANCAT-*L1*, 3) ANCAT-*Alg.Lat.*, and 4) ANCAT-*UPL* with ANCAT models having $K = 10$ and varying regularization term λ. With respect to perceived latency, ANCAT-*UPL* provides the superior accuracy-latency trade-off. (Right) ANCAT-*UPL* models trained with a fixed regularization factor but with different throughputs. Faster compute power gives the model more lookahead flexibility, resulting in better accuracy-latency operating points for ANCAT-*UPL* models.

E. Attention Dynamics

Plots are presented for frame-wise latency measures through an example utterance in Figure 12, and full attention masks for all layers on both LibriSpeech and Voice Assistant data examples are also displayed in Figures 13 and 14. Figure 12 shows that algorithmic latency can vary greatly over each frame of an utterance with spikes occurring in places where the model is less confident and desires further future context. We see similar spikes for the accumulation and burn down of compute backlog throughout the utterance. Note, however, that the rise and fall of the backlog is steadier throughout since progress completing nodes can more consistently occur as dependencies for intermediate layer nodes are met. Also observe, that in this 46-frame example (2.76 seconds long), because there should be no forward attention past the end of the utterance, the algorithmic latency is rightfully 0 at the utterance boundary and beyond. Likewise, because compute backlog defines the amount of work remaining and there is a non-zero backlog at the final frame, this utterance will observe a positive UPL. Minimizing this backlog at the utterance boundary (frame index 45) is the objective of our UPL loss algorithm.



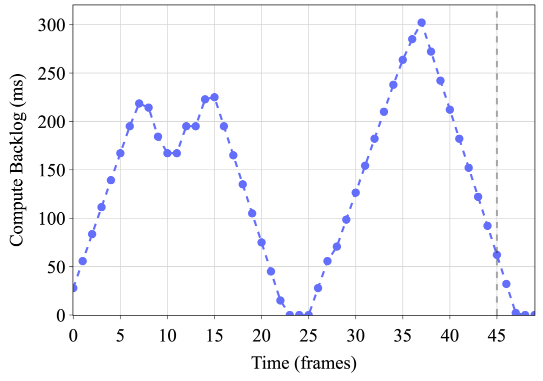


Figure 12. Voice Assistant Conformer ANCAT model frame-wise algorithmic latency (left) and the compute backlog present at each frame for an utterance (right). The utterance is 46 frames in length and a throughput of 2 is used for the compute backlog evaluations.

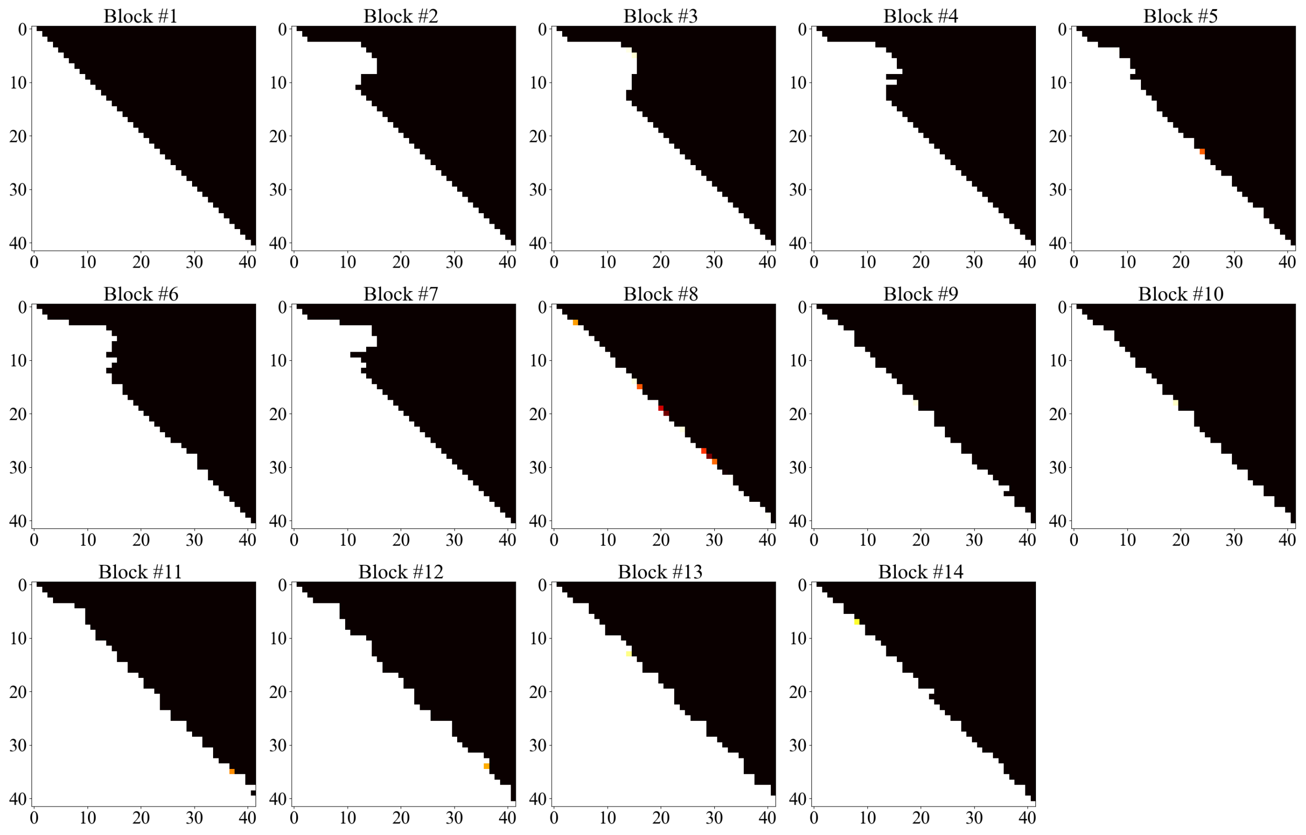


Figure 13. LibriSpeech example utterance block-wise attention masks for Conformer ANCAT-*Alg.Lat.* model. One observes the distinctive stepwise patterns of ANCAT-*Alg.Lat.* with large amounts of clustered lookahead at particular segments within the utterance.

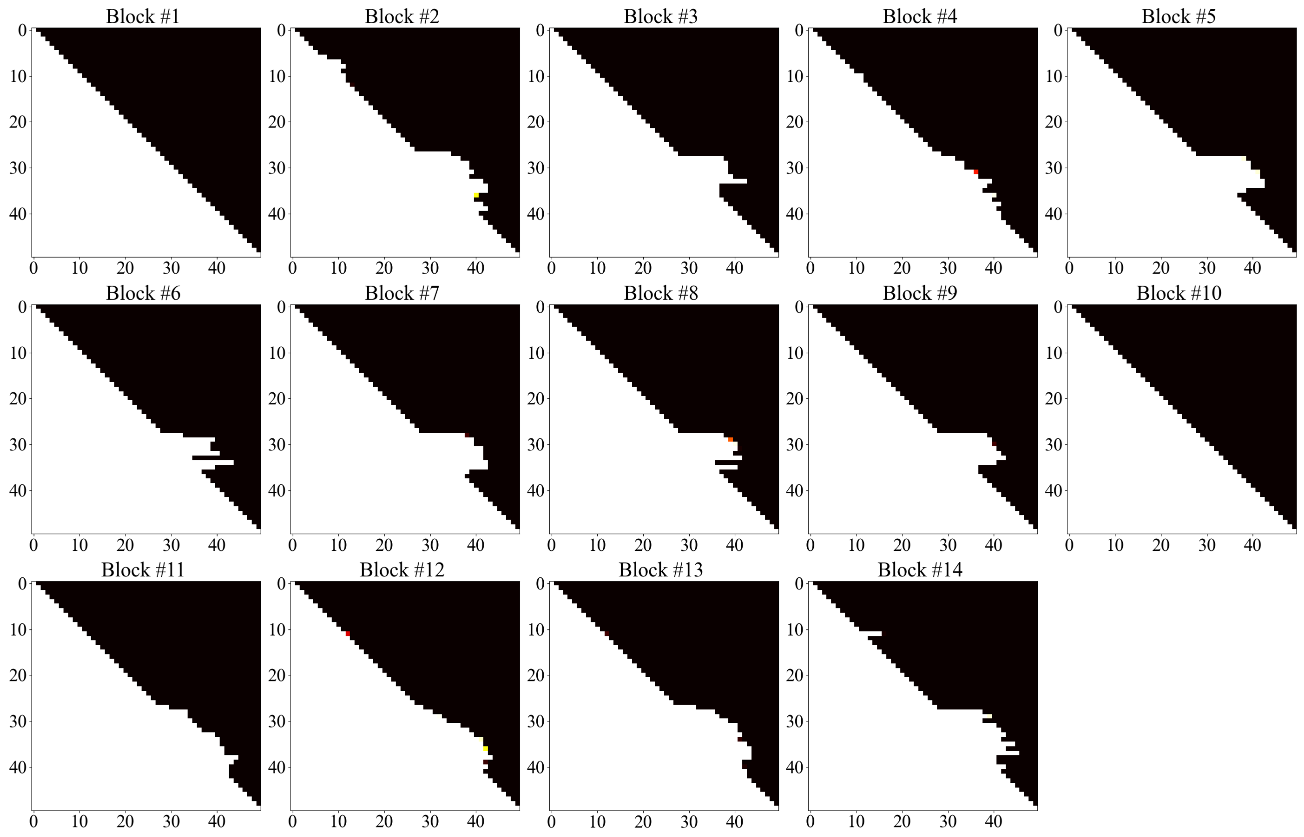


Figure 14. Voice Assistant example utterance block-wise attention masks for Conformer ANCAT-*Alg.Lat.* model. One observes the distinctive stepwise patterns of ANCAT-*Alg.Lat.* with large amounts of clustered lookahead at particular segments within the utterance.

F. Training Characteristics

Throughout training, the temperature parameter τ is annealed from 1 to near 0, which transitions the soft DAG edges (and attention mask) into binary ones. This appendix shows how the changes of τ throughout training impacts various components of the ANCAT architecture.

Figure 15 shows the various future frame scheduler masking values at different temperatures. Notice how the curve sharpens from a reverse “S” shape into a step function.

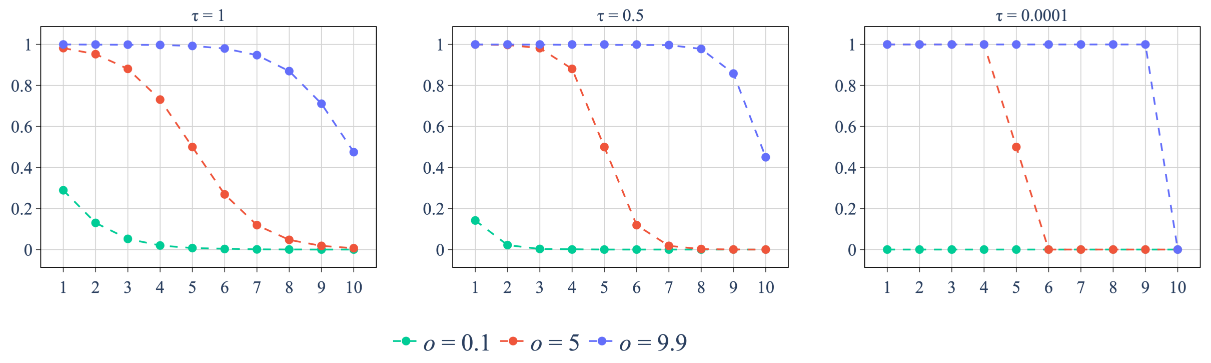


Figure 15. Scheduler values for different predicted curve centers o with a maximum span of $K = 10$ at various temperatures. These values mask the forward attention positions and represent the presence of a forward edge in the compute DAG. As temperature decreases, the edges become binarized.

During training, ANCAT has soft edges. The loss function therefore computes a corresponding "soft latency". Meanwhile, the "hard latency", where all non-zero edge values are treated as 1, is used to evaluate the true run-time latency at test time. Naturally these two measures are different at the start of training. We show in Figure 16 however, that as temperature anneals over time, hard and soft latency measures smoothly converge to the ultimate final algorithmic latency, with the hard always an upper bound on soft.

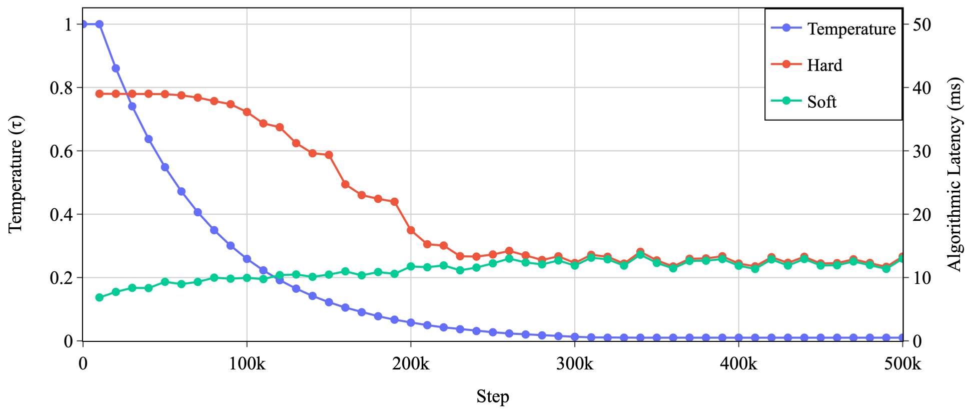


Figure 16. Temperature, soft latency, and hard latency over training. Model checkpoints were evaluated every 10k steps with a Conformer ANCAT-*Alg.Lat.* model.

Figures 17 and 18 visualize per-frame attention, algorithmic latency, and compute backlog for an ANCAT model at three checkpoints during training for the same 46-frame utterance from the test set. Figure 17 shows the evolution of the lookahead attention masking values for the utterance and how they evolve to their final position. One observes how the training begins with many fractional attention mask values but becomes more polarized (closer to 0 or 1) due to temperature annealing. For the last checkpoint, where the temperature τ is 0.01, attention masking is nearly binary.

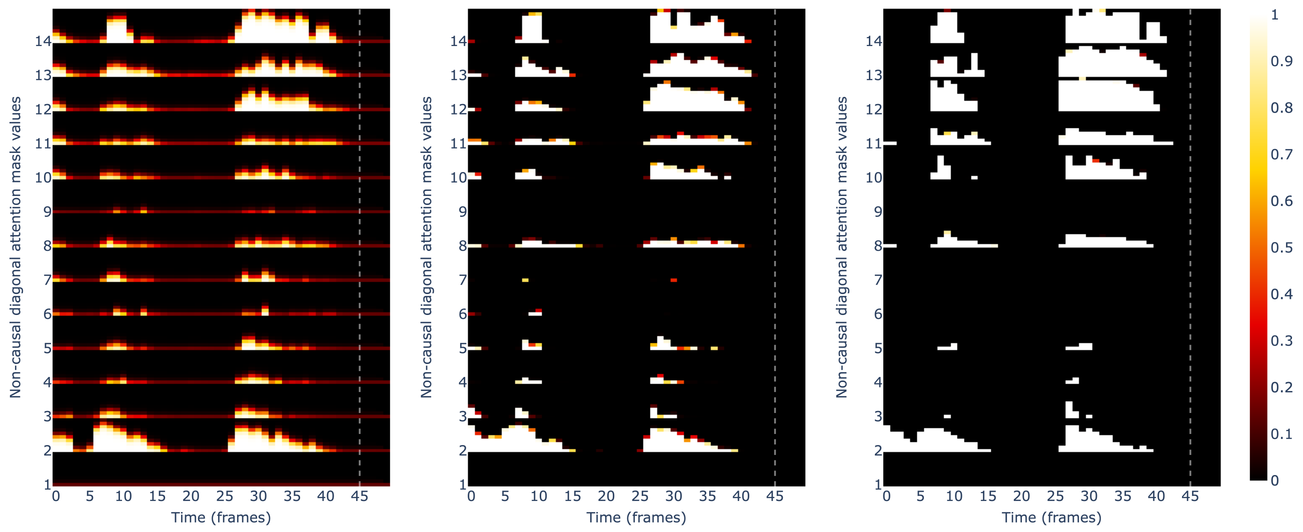


Figure 17. Per-frame non-causal attention mask values of an utterance for a Conformer ANCAT-*UPL* model. The mask values are derived from checkpoints evaluated at step 50k where temperature is 0.548 (left), at step 160k where temperature is 0.105 (center), and at step 500k, the final step, where temperature is 0.01 (right). The utterance consists of a far-field user speaking "[Voice Assistant], play Christmas music." The max forward span K of each layer is 10, and the figure shows only the 10 forward mask values right of the main diagonal of the full mask. The 10 forward mask values ascend from bottom to top. Layers are labeled on the left vertical axes. Throughout training, the attention masks shift from fractional to nearly binary. Most of the attention structure is preserved but with definite differences as the model converges and adjusts itself to the annealing temperature.

Next, we visualize the characteristics of per-frame soft and hard latency throughout training in Figure 18. First, observe that although per-frame soft and hard algorithmic latency patterns differ greatly at earlier epochs, they converge to nearly the same values at the final checkpoint as expected. See also in the first checkpoint that hard algorithmic latency is at its maximum, matching what one would expect from a full-context attention model, and similarly at this checkpoint, the backlog gradually builds over time as more work than can be executed in real-time is added frame over frame.

Next, comparing frame-wise latency between checkpoints, we see soft algorithmic latency keeping a similar shape, but the amplitude increases slightly over the course of training. This suggests that earlier in training, the model can learn to keep important attention weights small, yet non-zero, to minimize the soft penalty term. As temperature decreases however, these non-zero weights become increasingly polarized where the now stronger weighted forward connections cause higher algorithmic latency. This matches Figure 16 findings.

Finally, we discuss the per-frame compute backlog. Similar to the trends we saw with algorithmic latency, we see soft and hard latency converge over the epochs. Soft backlog maintains the same shape but increases in amplitude. The hard backlog starts at a maximum and then decreases to converge with the soft backlog calculation. Taking a closer examination of the hard backlog in the first checkpoint, one notices the backlog continues to grow frame by frame due to the large hard algorithmic latency. Close examination reveals that at every 10 frames, there is a slight decrease in slope. This phenomenon occurs because after 10 frames, the nodes for the first layer start to become available, since the max per-layer right context is 10 frames. After 20 frames, the second layer starts to become available, and so on every 10 frames. If the utterance were long enough, we would see that the hard compute backlog would reach its maximum value at 140 frames (14 layers $\times$ 10 lookahead).

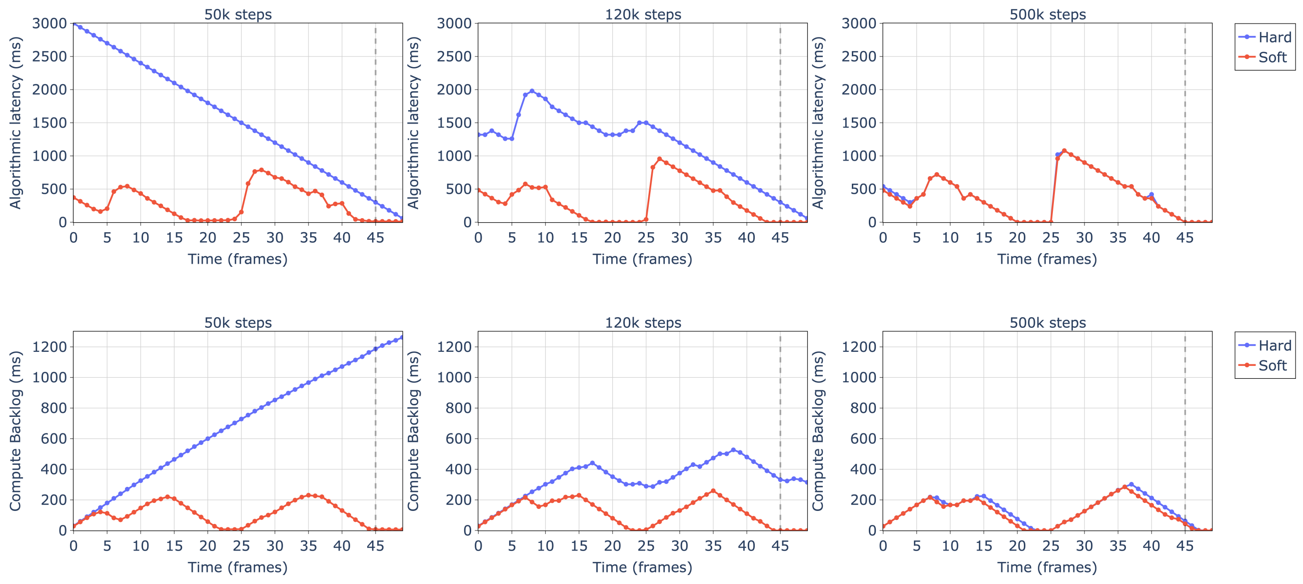


Figure 18. Voice Assistant Conformer ANCAT model frame-wise algorithmic latency (top) and the compute backlog present at each frame (bottom) for an utterance evaluated at three checkpoints during training: step 50k where temperature is 0.548 (left), step 160k where temperature is 0.105 (center), and step 500k the final step, where temperature is 0.01 (right). A throughput of 2 is used for the compute backlog evaluations.

G. Training Configuration Specifics

All the models presented in this paper on LibriSpeech and Voice Assistant data are trained on 6×8 NVIDIA Tesla V100 GPUs with a per-core bucket batch size of [32, 16, 12, 4] according to different utterance lengths. For our LibriSpeech ANCAT models, we initially set the temperature for schedulers as $\tau = 1$ and anneal it to $1\mathrm{e}{-4}$ with an exponential decay rate of 0.99996 beginning at epoch 40. Voice Assistant uses a longer decay annealing over 320k steps to a temperature of 0.01. We also list the detailed encoder configuration of the Conformer model in Table 3. The T-T architecture we use in the experiments has the same hyperparameters except those of the convolutions models which do not apply. The scheduler networks consist of a total of 0.2M parameters which are considerably lightweight compared to our main Transformer-based encoder.

Table 3. Encoder Model Hyperparameters for Conformer.

Conformer Encoder Setup	
Num of Params (M)	27.5
Encoder Layers	14
Encoder Dim	256
Num of Attention Heads	4
Feed-Forward Dim	1024
Point-wise-1 Conv Kernel Size	(512, 256, 1, 1)
Depth-wise Conv Kernel Size	(32, 1, 1, 1)
Point-wise-2 Conv Kernel Size	(256, 512, 1, 1)
Output Size	512

H. Voice Assistant Result Tables

For completeness, we also construct a table for Voice Assistant data as done with LibriSpeech for both Conformer and T-T baselines and ANCAT models. We provide model label names for ease of reference.

Label	Attention (right context frames)	Regularization Term	Reg. weight	Throughput	WERR	Alg. Latency (ms)	UPL (ms)
Baseline	Causal	N/A	N/A	2.0	0.0	0.0	0.0
Layer1	Layerwise (1)	N/A	N/A	2.0	-9.2	797.3	224.2
Layer2	Layerwise (2)	N/A	N/A	2.0	-11.9	1399.7	446.7
Layer10	Layerwise (10)	N/A	N/A	2.0	-21.1	2323.0	1160.8
Chunked5	Chunked (5)	N/A	N/A	2.0	-3.9	122.1	82.8
Chunked10	Chunked (10)	N/A	N/A	2.0	-8.2	277.5	194.7
Chunked20	Chunked (20)	N/A	N/A	2.0	-11.2	583.2	402.3
Chunked30	Chunked (30)	N/A	N/A	2.0	-12.0	871.4	681.6
L1-A	ANCAT (10)	L1	1.00e-04	2.0	-16.8	1519.0	772.5
L1-B	ANCAT (10)	L1	2.50e-04	2.0	-14.4	1177.9	606.8
L1-C	ANCAT (10)	L1	5.00e-04	2.0	-12.9	825.2	361.4
L1-D	ANCAT (10)	L1	7.50e-04	2.0	-11.2	619.4	239.8
L1-E	ANCAT (10)	L1	1.00e-03	2.0	-9.4	587.5	236.7
L1-F	ANCAT (10)	L1	1.25e-03	2.0	-8.7	412.7	150.0
L1-G	ANCAT (10)	L1	1.50e-03	2.0	-7.0	266.0	33.4
L1-H	ANCAT (10)	L1	1.75e-03	2.0	-5.8	242.1	21.9
L1-I	ANCAT (10)	L1	2.00e-03	2.0	-6.6	188.1	11.2
L1-J	ANCAT (10)	L1	2.50e-03	2.0	-5.2	206.8	22.4
AL-A	ANCAT (10)	Alg. Latency	1.25e-03	2.0	-17.1	1007.9	712.5
AL-B	ANCAT (10)	Alg. Latency	2.50e-03	2.0	-13.9	884.8	628.0
AL-C	ANCAT (10)	Alg. Latency	3.75e-03	2.0	-14.3	657.1	467.4
AL-D	ANCAT (10)	Alg. Latency	5.00e-03	2.0	-11.7	503.4	344.4
AL-E	ANCAT (10)	Alg. Latency	7.50e-03	2.0	-9.5	256.7	46.6
AL-F	ANCAT (10)	Alg. Latency	1.00e-02	2.0	-8.9	176.3	12.0
AL-G	ANCAT (10)	Alg. Latency	1.50e-02	2.0	-6.5	88.2	3.1
AL-H	ANCAT (10)	Alg. Latency	1.75e-02	2.0	-4.8	81.8	2.5
CB01.5-A	ANCAT (10)	UPL	8.75e-05	1.5	-14.8	888.1	264.5
CB01.5-B	ANCAT (10)	UPL	1.30e-04	1.5	-14.9	725.2	154.4
CB01.5-C	ANCAT (10)	UPL	1.93e-04	1.5	-14.0	611.8	107.6
CB01.5-D	ANCAT (10)	UPL	4.38e-04	1.5	-10.0	526.5	58.5
CB01.5-E	ANCAT (10)	UPL	9.77e-04	1.5	-9.9	455.9	22.8
CB2-A	ANCAT (10)	UPL	1.17e-04	2.0	-16.9	814.9	174.4
CB2-B	ANCAT (10)	UPL	1.73e-04	2.0	-17.9	796.0	125.6
CB2-C	ANCAT (10)	UPL	2.57e-04	2.0	-14.3	642.2	72.0
CB2-D	ANCAT (10)	UPL	5.83e-04	2.0	-13.4	585.2	47.6
CB2-E	ANCAT (10)	UPL	1.30e-03	2.0	-12.4	482.2	17.1
CB3-A	ANCAT (10)	UPL	1.75e-04	3.0	-17.4	890.0	108.4
CB3-B	ANCAT (10)	UPL	2.59e-04	3.0	-17.1	722.6	51.9
CB3-C	ANCAT (10)	UPL	3.85e-04	3.0	-14.7	673.6	48.2
CB3-D	ANCAT (10)	UPL	8.75e-04	3.0	-13.8	594.5	24.2
CB3-E	ANCAT (10)	UPL	1.95e-03	3.0	-12.3	550.6	12.2

Table 4. Voice Assistant Conformer Experimental Results.

Label	Attention (right context frames)	Regularization Term	Reg. weight	Throughput	WERR	Alg. Latency (ms)	UPL (ms)
Baseline	Causal	N/A	N/A	2.0	0.0	0.0	0.0
Layer10	Layerwise (10)	N/A	N/A	2.0	-18.1	2322.7	1160.9
Chunked10	Chunked (10)	N/A	N/A	2.0	-5.5	277.5	194.7
Chunked20	Chunked (20)	N/A	N/A	2.0	-9.5	583.3	402.4
L1	ANCAT (10)	L1	5.00e-04	2.0	-15.6	967.6	310.5
AL	ANCAT (10)	Alg. Latency	5.00e-03	2.0	-16.4	563.8	292.0

Table 5. Voice Assistant Transformer-Transducer Experimental Results.