TDNN-Conformer: Enhancing the Conformer with Time Delay Neural Networks

Zachary Zusin
Columbia University
zachary.zusin@columbia.edu

***Abstract*—This study introduces a novel modification to the Conformer architecture for automatic speech recognition (ASR) by replacing its convolutional modules with Time Delay Neural Networks (TDNNs). While Conformers have demonstrated remarkable success in ASR tasks through their combination of convolution and attention mechanisms, we hypothesize that incorporating TDNN layers could enhance both computational efficiency and recognition accuracy Our proposed architecture maintains the Conformer's powerful sequence modeling capabilities while leveraging TDNN's efficient temporal processing. We evaluate this model using the LibriSpeech dataset, and directly compare its performance to the original Conformer model.**

***Index Terms*—Automatic Speech Recognition, Time Delay Neural Networks, Transformers, Conformers, Deep Learning**

I. Introduction

SPEECH recognition presents unique challenges due to the multi-scale nature of spoken language. At the lowest level, systems must accurately convert acoustic signals into phonemes, a unit of sound in a specified language, requiring precise temporal processing at the millisecond scale. At higher levels, they must understand words and phrases in context, necessitating the ability to model relationships spanning seconds or even minutes. As a result, automatic speech recognition (ASR) systems must effectively combine local auditory information with broader linguistic context to make accurate predictions.

A. Conformers

Recent advances in ASR have been dominated by transformer-based architectures, with the Conformer emerging as a particularly effective approach. [1] At its core, the Conformer combines the global modeling capabilities of Transformers with the local processing abilities of Convolutional Neural Networks (CNNs). The self-attention mechanisms of Transformers have been shown to be extremely effective at capturing relationships between distant words, while CNNs are effective at detecting spatial patterns, particularly useful in analyzing visual inputs such as spectrograms (visual representations of the frequencies of a signal such as sound).

Conformers integrate three critical components: Multi-Headed Self-Attention (MHSA) modules, convolution modules, and feed-forward networks, arranged in a unique "macaron-style" configuration. The MHSA module enables the network to capture long-range dependencies across the input sequence, allowing each input to attend to all other inputs and create context-aware representations.

Simultaneously, the convolution module focuses on extracting local, fine-grained temporal and spectral patterns, providing a complementary perspective to the global attention mechanism. This architectural design allows Conformers to effectively maintain an understanding phenomena at many time scales, making them particularly powerful for processing complex sequential data like speech.

B. Time Delay Neural Networks

Independently, Time Delay Neural Networks (TDNNs) have maintained their relevance in modern ASR systems, particularly in the Kaldi toolkit where they have demonstrated impressive performance in phoneme classification tasks where precise temporal boundaries are difficult to determine. Their ability to identify acoustic features independent of their position in time has provided significant advantages over traditional static classification approaches and as a result has kept them at the forefront of the domain.

TDNNs are specialized neural networks designed for modeling temporal relationships in speech. Unlike traditional multilayer perceptrons that process each time step independently, TDNNs employ a hierarchical structure of time-delay connections. At each layer, neurons receive input not just from the current time step, but from a carefully chosen set of past and future time steps, creating what are called "contextual windows." [6]

This is implemented through dilated convolutions, where the spacing between input samples increases as you move up the network hierarchy. The dilation pattern allows TDNNs to exponentially expand their receptive field (the total amount of time they can analyze) without a proportional increase in computational complexity. This efficient architecture makes TDNNs particularly well-suited for real-time ASR applications where computational resources may be limited.

TDNNs also exhibit shift invariance, meaning they can recognize acoustic patterns without having to segment the signal into beginning and end points of distinct sounds. This property is crucial for ASR, as the same phoneme can occur at different speeds or positions within an utterance while maintaining its linguistic identity.

II. Formulation from State of the Art

The integration of TDNN architectures with Conformers presents a unique opportunity to enhance automatic speech recognition systems by combining their complementary

strengths. While Conformers effectively make use of their MHSA module's self-attention mechanism, their traditional convolutional components may not optimally capture the hierarchical temporal dependencies inherent in speech. Modern TDNN architectures, particularly those with factorized layers (TDNN-F) and dilated convolutions, offer several advantages that could address these limitations.

As was previously described, TDNNs are specifically designed for efficiently capturing local and medium-range temporal patterns critical for phoneme recognition and word boundary detection. TDNN-F layers can be implemented as an optimization of traditional TDNN layers where the weight matrix W is decomposed into the product of two matrices U and V such that $W = UV$. This factorization, combined with a semi-orthogonal constraint on U, $U^T U = I$, significantly reduces the number of parameters TDNNs used while maintaining model capacity. [4] The transformation at each time step t can be expressed as: $y_t = \sigma(U(Vx_{t+r}))$ where σ is a non-linear activation function, r represents the time delay, and x_{t+r} is the input vector at time $t + r$. This factorization typically reduces the parameter count by a factor of 2 to 3 while maintaining performance.

The position-independent processing capabilities of TDNNs align well with the position-dependent nature of self-attention mechanisms in Conformers. Unlike dense convolutions that process every temporal position with the same kernel, TDNNs use dilated connections that skip fixed intervals and so the sparsity inherent to the structure of TDNNs offers potential efficiency gains. These characteristics suggest that replacing the Conformer's convolution module with an appropriately designed TDNN module could enhance the model's performance while reducing computational overhead. The challenge lies in designing this integration to maintain the Conformer's powerful sequence modeling capabilities while leveraging TDNN's specialized construction.

III. Proposed Architecture

Our proposed architecture, the TDNN-Conformer, preserves much of the original Conformer encoder organization but replaces its convolutional module with a specialized TDNN module for improved temporal modeling. The architecture consists of an input processing pipeline, followed by a series of main processing blocks, and concludes with an output layer that produces the final predictions.

A. Overall Structure

The input processing stage begins with raw audio being converted into 80-dimensional log Mel filterbank features, a type of acoustic feature representation often used in speech recognition that approximate human auditory perception. These features undergo global layer normalization to standardize their distribution, preparing them for efficient processing by subsequent layers. The normalized features then pass through a linear projection layer that maps them to the model's internal dimension.

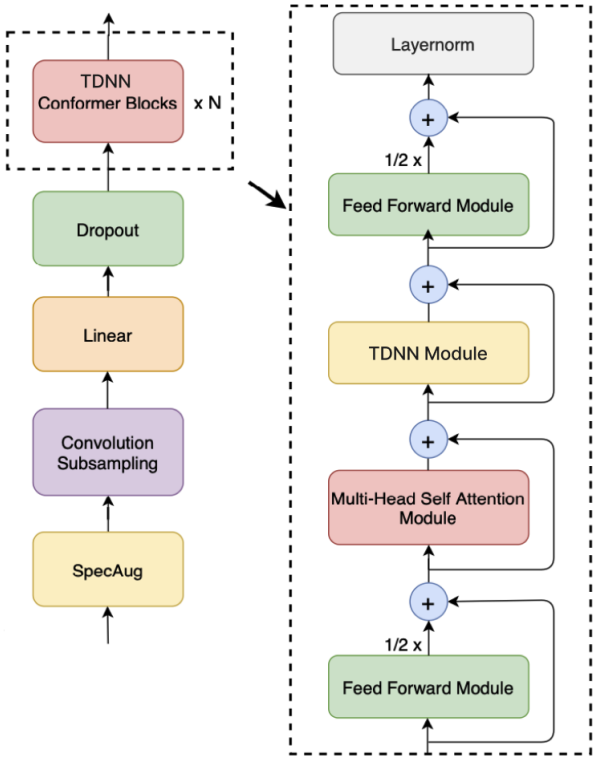


Figure 1: TDNN-Conformer encoder model architecture. The TDNN-Conformer comprises of two feed-forward modules with half-step residual connections surrounding the multi-headed self-attention and time delay neural network modules, followed by a layernorm layer.

The core of our architecture consists of N identical blocks, each containing four main components arranged in the original Conformer's "macaron-style" configuration. Two feed-forward networks (FFN) sandwich the multi-headed self-attention (MHSA) and TDNN modules. Each FFN module comprises two linear transformations with a SwiGLU activation function between them, expanding the features to four times the model dimension before projecting back to the original size. A dropout rate of 0.1 is applied for regularization, and the output is scaled by a factor of 0.5 to maintain stable gradients.

B. Multi-headed Self-attention Module

The second component in each block is the multi-headed self-attention module (MHSA), which remains unchanged from the original Conformer design. This module employs eight attention heads, each operating with a dimension of 1/8 of the total model dimension. The MHSA uses relative positional encoding instead of absolute positions, allowing it to better model the relative timing relationships in speech. A dropout rate of 0.1 is applied to the attention weights for regularization.

C. TDNN module

The most significant innovation in our architecture is the TDNN (Time Delay Neural Network) module, which replaces the original Conformer's convolution module. This component is structured with initial pointwise convolution followed by a Gated Linear Unit block that provides adaptive gating to control information flow. The core processing occurs through three parallel dilated depthwise convolutions, where each layer operates with progressively increasing dilation rates to efficiently capture wider contexts. After the TDNN layers process the input, their outputs are concatenated and passed through a pointwise convolution that learns to merge these multi-scale temporal features. Lastly, layer normalization is used to standardize feature distributions, followed by an activation function to introduce non-linearity. A final pointwise convolution processes this normalized and activated representation to produce the output.

D. Integration of Modules

The integration of components within each block is carefully managed through residual connections and layer normalization. Before each major component (MHSA, TDNN, and FFNs), layer normalization is applied to the input features. Residual connections bypass each component, allowing the model to maintain access to lower-level features. The output from all N blocks feeds into a final layer normalization followed by a linear transformation that produces logits for each token in our character vocabulary.

IV. Dataset and Experimental Setup

A. Data

Our experiments use the LibriSpeech corpus, a widely-used benchmark dataset for speech recognition derived from audiobook recordings. [3] The dataset contains 960 hours of speech data divided into training sets of 'clean' speech (460 hours) and 'other' speech (500 hours), representing optimal and more challenging recording conditions respectively.

In this study, however, given the limited computational resources we had access, a subset of the LibriSpeech dataset was used. In particular, the train-clean-100 portion, which contains 100 hours of high-quality read speech data, was trained on. This subset provided a reasonable balance between computational feasibility and dataset size needed to evaluate the architectural modifications made.

For evaluation, we used the standard dev-clean and test-clean sets, each containing approximately 5 hours of speech. We focused on the 'clean' test sets, audio with clear and unambiguous pronunciations of words, as they provide a fair evaluation of our model given the data it was trained on.

B. Data Preprocessing

The data preprocessing follows the WeNet framework implementation [7]. First, we create mappings from utterances to their corresponding audio files and transcriptions. Then, the raw audio waveform is transformed into log Mel filterbank features.

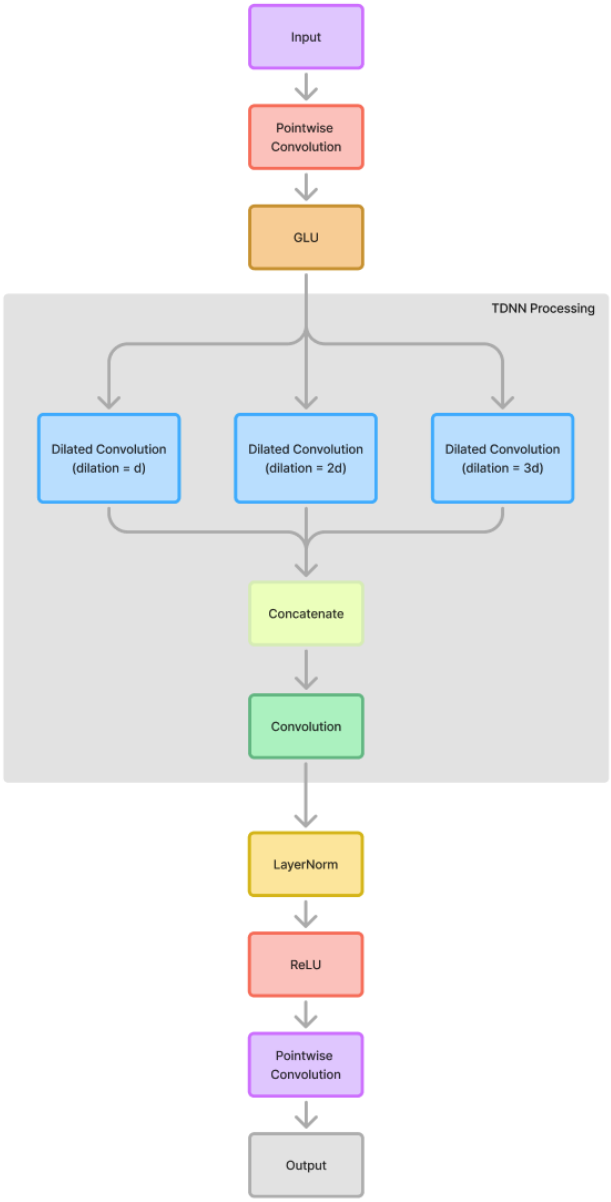


Figure 2: Architecture of the Time Delay Neural Network (TDNN) Module. The input features are processed through an initial pointwise convolution followed by a Gated Linear Unit (GLU). The core TDNN processing consists of parallel dilated convolutions, each operating with increasing dilation rates (d, 2d, 3d) and using groupwise convolutions with a shared kernel size. The outputs from these dilated convolutions are concatenated and combined through a 1×1 convolution. The final stages include layer normalization, ReLU activation, and a pointwise convolution.

This transformation involves dividing the signal into overlapping frames of 25ms duration, with each subsequent frame shifted by 10ms. This overlapping window approach ensures smooth capture of dynamics over time while maintaining sufficient resolution. For each frame, the power spectrum is computed and filtered using 80 triangular filters spaced according to the Mel scale, which emphasizes lower frequencies similar to human hearing. The logarithm is then applied to these filterbank energies, as human perception of sound intensity follows a logarithmic scale.

Global Cepstral Mean and Variance Normalization (CMVN) is then applied to these features to reduce acoustic variability caused by different recording conditions and speakers. CMVN works by calculating the mean and variance of the features across the entire training set and using these statistics to normalize the features to zero mean and unit variance. This normalization helps the model focus on the relevant acoustic patterns rather than speaker-specific or recording-specific variations, making the system less sensitive to these sorts of variations.

For text processing, we employ BPE (Byte Pair Encoding) tokenization with a vocabulary size of 5000, including special tokens for CTC blank symbol, unknown token, and start/end of sequence markers. BPE is a data compression technique adapted for text tokenization that iteratively merges the most frequent pairs of characters into single tokens. This creates a vocabulary that can efficiently represent both common words as single units and rare words as sequences of subword unit. This approach is particularly effective for our speech recognition task as it can handle out-of-vocabulary words by decomposing them into meaningful subword units.

The CTC (Connectionist Temporal Classification) blank symbol is a special token used in the CTC loss function, which enables the model to learn the alignment between speech frames and text tokens without requiring explicit frame-level labels. It allows the model to predict "no output" at certain frames, which is crucial because speech features are typically sampled at a much higher rate than text tokens. This helps handle the disparity between speech and text, as a single phoneme or word can span multiple audio frames.

Given a speech utterance of T frames and a corresponding text sequence of N tokens, the CTC loss allows all possible alignments between these sequences. Mathematically, for an input sequence X and target sequence Y, the CTC loss is computed as:

$$\mathcal{L}_{\text{CTC}} = -\log P(Y|X) = -\log \sum_{\pi \in \mathcal{B}^{-1}(Y)} P(\pi|X)$$

where π represents a possible alignment, $\mathcal{B}(Y)$ is the CTC mapping function that maps each alignment to a word (removing blanks and repeated characters), and $\mathcal{B}^{-1}(Y)$ is the set of all possible alignments that reduce to Y. [2]

C. Model Configuration

Our TDNN-Conformer implementation's hyperparameters were chosen to create a lightweight model that could run effectively on limited computational resources while maintaining competitive performance. The model uses 6 encoder blocks compared to the 16 blocks used in Conformer (M), but has a model dimension of 256 and 4 attention heads matching Conformer (M). For the TDNN module, we use a kernel size of 3, base dilation rate of 1, and context size of 2, which contrasts with the original Conformer's convolution module which used a larger kernel size of 32.

Model	TDNN-Conformer	Conformer (S)	Conformer (M)	Conformer (L)
Num Params (M)	12.5	10.3	30.7	118.8
Encoder Layers	6	16	16	17
Encoder Dim	256	144	256	512
Attention Heads	4	4	4	8
Conv Kernel Size	3	32	32	32
Decoder Layers	6	1	1	1
Decoder Dim	2048	640	320	640

Table 1: *Model hyper-parameters for the TDNN-Conformer, and the three original Conformer models from the Conformer: Convolution-augmented Transformer for Speech Recognition paper.*

D. Training Protocol

The training protocol follows WeNet's framework as well, using the AdamW optimizer with β_1=0.9, β_2=0.98, and gradient clipping with maximum norm 10.0. The learning rate schedule includes a 25,000-step linear warmup, reaching a peak rate of 0.001, followed by the standard Transformer learning rate scheduler, as specified by the following expression:

$$d_{\text{model}}^{-0.5} \cdot \min(\text{step\_num}^{-0.5}, \text{step\_num} \cdot \text{warmup\_steps}^{-1.5})$$

where d_{model} is the model dimension, step_num is the current training step, and warmup_steps is the number of warmup steps. [5]

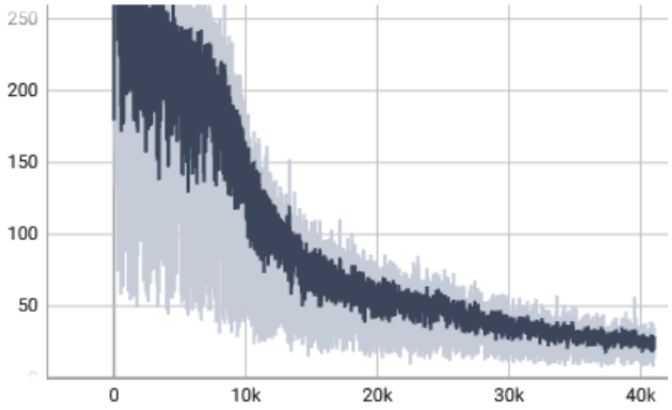
This scheduler has two distinct phases: (1) a warmup phase where the learning rate increases linearly from near zero to its peak value, and (2) a decay phase where the learning rate decays proportionally to the inverse square root of the step number. This scheduling strategy helps stabilize early training by gradually increasing the learning rate, then slowly decreases it to help fine-tune the model's parameters. The gradual warmup is particularly important for transformer architectures as it helps prevent excessive parameter updates in the multi-head attention layers during the initial training phase when gradients can be unstable.

We apply data augmentation during training using SpecAugment with reduced parameters: two frequency masks with maximum width 10 and two time masks with maximum width 50. SpecAugment is a data augmentation technique that randomly masks portions of the spectrogram in both time and frequency dimensions. The frequency masks remove frequency bands to help the model become more robust to variations in vocal tract characteristics, while the time masks remove segments of time to help the model handle variations in speaking speed. Additionally, we use speed perturbation for data augmentation as specified in the configuration. Training with these artificially corrupted inputs improves the model's resilience to real-world variations in speech.

Training proceeds with a batch size of 12 utterances on a single Nvidia RTX 2070 GPU with 32GB RAM for 35 hours, with model checkpoints saved every 1,000 steps. During training, we monitor several losses on the training set and afterwards performance is primarily measured using Word Error Rate (WER) on the test-clean set.

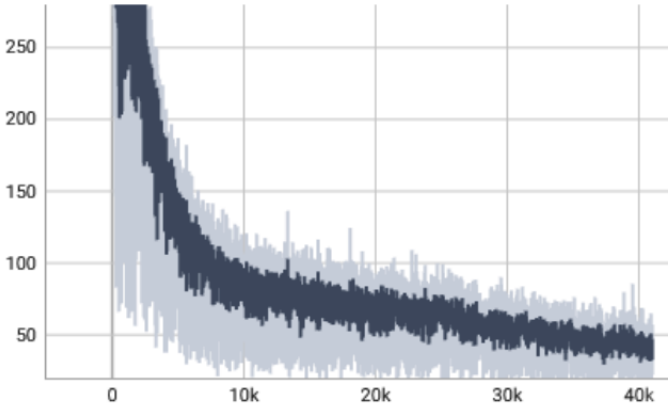
V. Results

A. Attention Loss



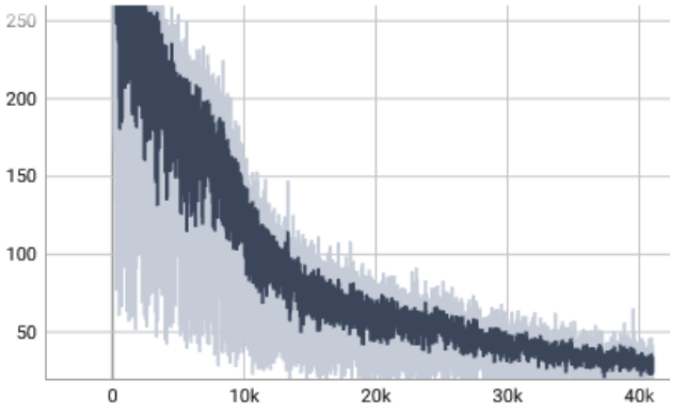
The attention mechanism's loss curve, which uses a label-smoothed cross-entropy loss comparing the model's predicted token distribution against smoothed targets (90% on correct label, 10% distributed across others), demonstrates the model's learning progression in attention-based speech recognition. Starting at approximately 250, the loss shows a sharp initial descent in the first 15,000 steps, followed by a more gradual decline through steps 15,000-35,000. This pattern indicates effective learning of the attention weights, with the model becoming increasingly effective at focusing on relevant parts of the input sequence for transcription while maintaining appropriate uncertainty through label smoothing.

B. Connectionist Temporal Classification Loss



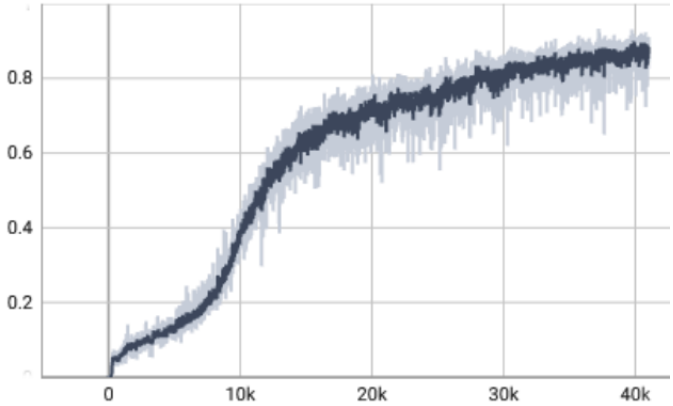
The Connectionist Temporal Classification (CTC) loss shows similar characteristics to the attention loss but with generally lower absolute values. Beginning at approximately 250, it demonstrates rapid improvement in the first 10,000 steps, followed by more a gradual decrease afterwards. The CTC loss stabilizes around 50 by the end of training, indicating effective learning of the alignment between input audio features and output text sequences.

C. Overall Training Loss



The combined training loss, which integrates CTC loss (30%) and attention-based label smoothing loss (70%), provides a comprehensive view of the model's overall performance. Starting from around 250, it shows a sharp initial decline followed by a steady decrease converging to values around 40-50. The consistent downward trend without significant plateaus or fluctuations suggests stable learning across both the CTC and attention components of the model. This smooth convergence indicates that both the alignment learning through CTC and the sequence modeling through attention are successfully being optimized, with the 0.3/0.7 weighting providing a balanced training signal. The label smoothing factor of 0.1 appears to be providing appropriate regularization, helping prevent overfitting while allowing the model to maintain strong predictive performance. The consistent downward trend without significant plateaus or fluctuations suggests stable learning across all components of the model and indicates successful minimization of the training objective.

D. Token-level Accuracy



The accuracy curve shows the model's improvement in token prediction over time. Initially, there is a period of slow

improvement from steps 0 to 5,000, followed by rapid learning until around step 15,000 where accuracy increases dramatically from 15% to 60%, and finally a phase of continued but more gradual improvement, reaching approximately 90% accuracy on the training dataset after 35 hours. This sigmoid-shaped learning curve is characteristic of successful model training, indicating effective knowledge acquisition without significant overfitting.

E. Word Error Rate

Model	Word Error Rate (%)
TDNN-Conformer	14.6
Conformer (S)	2.7
Conformer (M)	2.3
Conformer (L)	2.1

Table 2: *Comparison of Word Error Rates of the TDNN-Conformer and the Conformer models.*

Word Error Rate (WER) is calculated by dividing the sum of substitutions, insertions, and deletions by the total number of words in the transcript, and it is the primary metric used to measure how accurate a speech recognition is. Our trained model was applied on the input acoustic feature sequences using four different methods. Using the *CTC Greedy Search Method*, the most likely label is chosen at each time step without considering future context and a WER of 20.90% was calculated. In the *CTC Prefix Beam Search* method, multiple candidate sequences are tracked and expanded, improving accuracy through broader hypothesis exploration and a WER of 14.63% was found. The *attention* approach focuses on different parts of the input to guide the decoder's predictions through a learned alignment mechanism and had a WER of 20.78%. With the *Attention Rescoring* decoding, the attention-based decoder re-ranks candidates generated by *CTC Prefix Beam Search* to refine the final output and achieved a WER of 18.28%. Although these results, the lowest of which being a word error rate of roughly 14%, are substantially worse than those of Conformer models, which achieve scores as low as 2.7%, future work will likely bring these figures significantly down.

VI. Conclusion and Future Work

A. Summary of Results

This study introduces and evaluates a novel modification to the Conformer architecture, incorporating Time Delay Neural Network modules to enhance temporal modeling capabilities in speech recognition tasks. Despite the limitations imposed by computational constraints, our experimental results suggest the TDNN-Conformer performs very well in the speech recognition task and still has potential to improve.

B. Model Size and Computational Resources

The most obvious direction for future research lies in implementing and evaluating a full-sized TDNN-Conformer and using significantly more computational resources. A larger model on par with the Conformer (L) model paired with increased training time would enable a much more direct comparison with state-of-the-art models on standard benchmarks and provide a more complete understanding of the architecture's capabilities. Rather than use a single GPU and train on the scale of hours as done in this study, a more complete approach would make use of multiple GPUs and train on the order of days. Only then would it be possible to truly assess whether the performance of the TDNN-Conformer continues to scale with more training as the Conformer does or reaches a "ceiling" in its speech recognition abilities.

C. Language Models

Additionally, language model integration represents another way in which the TDNN-Conformer model can be tested. While our current implementation focused only on acoustic model components, incorporating external language models could significantly improve recognition accuracy. In their paper *Conformer: Convolution-augmented Transformer for Speech Recognition*, Anmol Gulati et al were able to go from having a WER of 4.3% without using language models to a WER of 3.9% with an external language model with their largest 118M parameter model. However, even without incorporating a language model, they were able to outperform the best known Transformer, LSTM, or convolution model with their medium sized model, a result the TDNN-Conformer has yet to achieve. Nonetheless, it would still be useful to investigate whether the TDNN-Conformer model would exhibit similar improvements or remain stagnant.

D. TDNN Module Configuration

Different configurations of TDNN layers, including variations in context window sizes and the potential implementation of factorized TDNN (TDNN-F) designs, could improve performance or efficiency. While this study implemented basic TDNN layers, exploring factorized versions that decompose the weight matrix into the product of two smaller matrices with semi-orthogonal constraints could significantly reduce the parameter count while maintaining model capacity. This study only evaluated one particular TDNN structure, so naturally it would be reasonable to investigate how the model performs with differently designed Time Delay Neural Networks. Also, the optimal balance between TDNN and attention mechanisms requires further study, as different configurations might be more effective for different types of speech recognition tasks.

E. Dataset

Lastly, just as training time and model capacity can be increased in future research, the amount and variety of data the model is trained on can also be increased as well. Firstly, using the entire full 960 hour long LibriSpeech dataset would of course provide more data to train, but also incorporating

the 500 hours of "non-clean" speech not used in this study would likely allow the model perform well in more realistic environments. Also, although we made use of data augmentation, particularly with SpecAugment, future data processing pipelines may see improved results in applying other data augmentation techniques such as speed perturbation, in which several versions of each training audio are created by adjusting the original recordings playback speed. Evaluating the architecture's effectiveness across different speech recognition tasks and domains, including multi-lingual speech recognition, noisy environment performance, and real-time applications would provide valuable insights into its practical utility.

And so, while the results of this preliminary study were constrained, they still suggest that the TDNN-Conformer architecture may represent a promising direction for future study. Once a full-scale implementation of the model is fine-tuned, we anticipate that this new architecture will contribute to the advancement of speech recognition technology.

REFERENCES

[1] Anmol Gulati et al. *Conformer: Convolution-augmented Transformer for Speech Recognition*. 2020. arXiv: 2005.08100 [eess.AS]. URL: https://arxiv.org/abs/2005.08100.

[2] Kalpesh Krishna et al. *A Study of All-Convolutional Encoders for Connectionist Temporal Classification*. 2018. arXiv: 1710.10398 [cs.CL]. URL: https://arxiv.org/abs/1710.10398.

[3] Vassil Panayotov et al. "Librispeech: An ASR corpus based on public domain audio books". In: *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2015, pp. 5206–5210. DOI: 10.1109/ICASSP.2015.7178964.

[4] Daniel Povey et al. "Semi-Orthogonal Low-Rank Matrix Factorization for Deep Neural Networks". In: *Interspeech 2018*. 2018, pp. 3743–3747. DOI: 10.21437/Interspeech.2018-1417.

[5] Ashish Vaswani et al. *Attention Is All You Need*. 2023. arXiv: 1706.03762 [cs.CL]. URL: https://arxiv.org/abs/1706.03762.

[6] A. Waibel et al. "Phoneme recognition using time-delay neural networks". In: *IEEE Transactions on Acoustics, Speech, and Signal Processing* 37.3 (1989), pp. 328–339. DOI: 10.1109/29.21701.

[7] Zhuoyuan Yao et al. "WeNet: Production oriented Streaming and Non-streaming End-to-End Speech Recognition Toolkit". In: *Proc. Interspeech*. IEEE. Brno, Czech Republic, 2021.