Clairaudience Team's Speech Enhancement System

This report details the development of the Clairaudience Team's neuromorphic speech enhancement system for the Intel Neuromorphic Deep Noise Suppression (Intel N-DNS) Challenge. The system introduces an innovative spiking neuron-based full-band and sub-band fusion model named N-FullSubNet for real-time speech enhancement. The architecture of N-FullSubNet stands as a beacon of creative engineering, comprised of two primary elements. The system's heart is driven by the novel Gated Spike Unit (GSU), a state-of-the-art spiking neuron model. This central powerhouse is neatly encapsulated within an enhanced version of the FullSubNet model, which now encompasses frequency partitioning. This advanced modification not only amplifies the system's capacity for speech enhancement but also significantly elevates its computational efficiency.

A. Gated Spiking Neuron Model

Despite incorporating recurrent dynamics, the Leaky Integrate-and-Fire (LIF) neuron model struggles to achieve high performance in speech enhancement tasks. This is mainly due to the fixed decay factor $\lambda \in \mathbb{R}$ for every neuron, which restricts their ability to modify the retained membrane potential flexibly. A recently proposed Parametric LIF (PLIF) replaces the fixed λ with learnable parameters that pass through the sigmoid function $\sigma(\boldsymbol{\lambda}) \in \mathbb{R}^N$. However, it still falls short as the decay factor remains constant across different time steps. To overcome this limitation, we introduce a gating function to regulate the decay rate at each time step. This allows each neuron to dynamically adjust its membrane potential, strengthening its capability to process temporal tasks. The neuronal dynamics of GSN can be formally expressed as follows:

$$\boldsymbol{i}^l[t] = \boldsymbol{W}_{mn}\boldsymbol{o}^{l-1}[t] + \boldsymbol{W}_{nn}\boldsymbol{o}^l[t-1] + \boldsymbol{b} \tag{1}$$

$$\boldsymbol{\lambda}^l[t] = \sigma(\boldsymbol{W}_{mn}\boldsymbol{o}^{l-1}[t] + \boldsymbol{W}_{nn}\boldsymbol{o}^l[t-1] + \boldsymbol{b}) \tag{2}$$

$$\boldsymbol{u}^l[t] = \boldsymbol{\lambda}^l[t]\boldsymbol{u}^l[t-1] + (1-\boldsymbol{\lambda}^l[t])\boldsymbol{i}^l[t] \tag{3}$$

When the membrane potential surpasses a predefined threshold, an output spike is triggered, followed by a resetting process. We intentionally utilize the same weight matrices for $\boldsymbol{\lambda}^l[t]$ as those used in calculating the input current as described in Equation (1). As a result, our proposed GSN model has the same number of parameters as PLIF.

B. FullSubNet with Frequency Partitioning

FullSubNet is a powerful model that synergistically combines a full-band model and a sub-band model, executing joint optimization. In this framework, the full-band model gleans global spectral information and extensive cross-band dependencies, while the sub-band model independently processes frequency bands, emphasizing local spectral patterns, reverberation characteristics, and signal stationarity. Experimental evidence supports the effective integration of these two complementary models within a single framework. However, FullSubNet's Achilles' heel lies in the computationally intensive sub-band component, which processes each band individually. This method contrasts with the human auditory system's differential processing of frequencies. For instance, mid-high frequency bands may not necessitate granular processing, while low-frequency bands might benefit from meticulous processing, and several high-frequency bands could be processed concurrently. Addressing this, we introduce a frequency partitioning technique. This approach applies different processing granularities to frequency bands, mirroring the human auditory system's variable frequency handling. This not only reduces computational demand but also maintains performance levels, as confirmed by preliminary experiments. Moreover, frequency partitioning allows for tailored processing, with more deep filtering applied to the better temporally-correlated low-frequency bands and less to high-frequency bands. This method refines the FullSubNet model, enhancing its efficiency without compromising performance.

C. Multiframe Deep Filtering

Traditional models for auditory masking estimation typically calculate each time-frequency mask independently, thus ignoring the inherent correlations across adjacent points in both time and frequency domains. Deep filtering resolves this issue by integrating context from neighboring points when determining the auditory masking for a specific time-frequency point. In comparison, while deep filtering slightly increases the computational load and parameter count relative to traditional methods, these increases are limited primarily to changes in the output layer. When integrated into the N-FullSubNet model, deep filtering facilitates the capture of more complex data patterns without significantly ramping up computational demands.

D. Loss Function Optimized with Black-Box Metrics

Our model's training employs a blend of loss functions for optimized effectiveness. First, we use the Scale-Invariant Signal-to-Distortion Ratio (SI-SDR) loss function $\mathcal{L}_{\text{SISDR}}$ for alignment consistency in the time domain. Then, we incorporate complex and magnitude spectrograms for robust frequency-level optimization. In recent times, adversarial learning with Generative Adversarial Networks (GANs) has proven efficacious in various applications. We include a MetricGAN+ discriminator to predict the Deep Noise Suppression Mean Opinion Score (DNSMOS), a perceptual metric miming human auditory impressions of speech quality.

$$\mathcal{L} = \alpha(100 - \mathcal{L}_{\text{SISDR}}) + ||\hat{S}(t,f)|^p - |S(t,f)|^p| + |\hat{S}(t,f) - S(t,f)| + \beta\mathcal{G} \tag{4}$$

where α and β are hyperparameters that balance the SI-SDR loss, frequency loss, and generator loss. p is the ratio of dynamic range compression.

E. Implementation Details

TOML is a widely used configuration file format within the Python community. We adopt TOML files for configuring our experiments. TOML configuration file contains various information, such as the type and parameters of the optimizer, the learning rate decay scheduler, details on gradient clipping, early stopping configurations, STFT parameters, and more. We developed four model variants, categorized by size: small, middle, large, and extra-large. Although they share the same fundamental architecture, these models differ primarily in the number of central frequencies, the frequency partitioning, the hidden sizes of the full-band and sub-band models, the deep filtering order, and the use of shared gates. The parameters for each model can be found in the corresponding TOML configuration file.

F. How to Training and Inference

Our training framework is built on the "core" + "recipes/dataset_name/model_name" directory structure, commonly adopted by renowned frameworks such as Kaldi and ESPnet. The "core" directory houses the common components used across various models. The recipes directory, on the other hand, contains sub-directories named after the specific datasets and models. These sub-directories store the unique scripts and configurations required to train and evaluate each specific model on the designated dataset. We provide comprehensive documentation[1] that covers all aspects of the training framework. This includes detailed instructions on setting up dependencies, an overview of the directory structure, an introduction to the experiment parameters, and a guide on how to run experiments.

G. How to Run Lava Inference

Despite numerous attempts, we still face persistent issues executing the foundational Lava inference code. These challenges prevent us from further integrating the Lava platform and our N-FullSubNet model. However, we maintain our optimism about the efficiency of our N-FullSubNet model on neuromorphic chips such as Loihi 2. Our future work will focus on resolving the technical issues hindering the implementation of the Lava inference. Once overcome, we hope to demonstrate the efficiency of our N-FullSubNet model on neuromorphic chips, further pushing the boundaries of what is possible in speech enhancement tasks.

H. Metricsboard

Entry	**SI-SNR** (dB)	**SI-SNRi**		**DNSMOS**			**Latency**		**Power proxy** (M-Ops/s)	**PDP proxy** (M-Ops)	**Param count** ($\times 10^3$)	**Model size** KB
		data (dB)	enc+dec (dB)	OVR	SIG	BAK	enc+dec (ms)	total (ms)				
Noisy	7.37	-	-	2.44	3.16	2.69	-	-	-	-	-	-
Small	13.89	6.52	6.52	2.97	3.28	3.93	0.03	32.03	29.24	0.94	521	2084
Middle	14.71	7.34	7.34	3.05	3.35	3.97	0.03	32.03	53.60	1.72	953	3816
Large	14.80	7.43	7.43	3.03	3.33	3.96	0.03	32.03	74.10	2.37	1289	5156
Extra Large	15.20	7.83	7.83	3.07	3.37	3.99	0.03	32.03	55.91	1.79	1798	7192

[1]Documentation: https://haoxiangsnr.github.io/audiozen/index.html