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 proposed system includes a novel SNN-based full-band and sub-band fusion model named Spiking-FullSubNet for real-time speech enhancement. The architecture of Spiking-FullSubNet stands as a beacon of creative engineering, comprised of two primary elements. The system's heart is driven by a novel Gated Spiking Neuron (GSN), a state-of-the-art spiking neuron model designed for speech enhancement. This central powerhouse is neatly encapsulated within an enhanced version of the FullSubNet model, along with a set of meticulously designed loss functions. This advanced modification not only amplifies the system's capacity for speech enhancement but also significantly elevates its computational efficiency.

A. Gated Spiking Neuron (GSN)

In our experiments, 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 $\lambda \in \mathbb{R}$ used for every neuron, which restricts their ability to retain multi-scale temporal information that is critical for speech enhancement. A recently proposed Parametric LIF (PLIF) [1] replaces the fixed λ with learnable ones, whose values are regulated via a 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. To save parameters, we reuse the same weight matrices for calculating $\boldsymbol{\lambda}^l[t]$ as those used in Equation (1). As a result, our proposed GSN model has the same number of parameters as PLIF [1].

B. FullSubNet with Frequency Partitioning

FullSubNet [2] is a popular speech enhancement model that synergistically combines a full-band model and a sub-band model. In FullSubNet, 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 at the same frequency granularity. This approach contrasts with the human auditory system which is more sensitive to low-frequency sounds. Addressing this, we introduce a frequency partitioning technique. This approach applies different processing granularities across the frequency bands, mirroring the human auditory system. Specifically, 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 refinement to the FullSubNet model not only reduces computational demand but also maintains performance levels, as confirmed in our experiments.

C. Multiframe Deep Filtering

For auditory mask estimation, traditional speech enhancement models typically calculate each time-frequency mask independently, thus ignoring the inherent correlations across adjacent points in both time and frequency domains. Deep filtering [3] 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 Spiking-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 loss function $\mathcal{L}_{\text{Freq}}$ on 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+ [4] discriminative loss $\mathcal{L}_{\text{Gen}}$ to predict the Deep Noise Suppression Mean Opinion Score (DNSMOS), a perceptual metric miming human auditory impressions of speech quality.

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

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

We adopt TOML files, a widely used configuration file format, 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 speech processing libraries such as Kaldi [5] and ESPnet [6]. 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[2] executing the provided official Lava inference code. These challenges prevent us from further integrating our Spiking-FullSubNet model with the Lava platform. 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 Spiking-FullSubNet model on Loihi chips, further pushing the boundaries of what is possible in speech enhancement tasks.

H. How you calcuated the latency for your solution?

To calculate the latency of our solution, we considered the following components and summed them up:

- **Data buffer latency**: We used STFT with a window shift of 8ms and a window length of 32ms. Since the STFT includes an overlap-add operation [7], the data buffer latency amounts to 32ms.
- **Encoder-decoder latency**: Our STFT configuration is the same as the official baseline [8], so we directly quoted the official result, i.e., 0.03ms.
- **Network latency**: Referring to the official paper [8] and accompanying documentation, we discovered that "Loihi 2 circuits typically complete all spike processing and neuron evaluations for a timestep within microseconds." Consequently, the network latency can be considered negligible at 0ms.
- **Look-ahead frames**: Our solution does not utilize any look-ahead frames or temporal convolutions,
- **Operations occur across time**: Our solution has no operations that occur across time.

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

[2]Github issue: https://github.com/IntelLabs/IntelNeuromorphicDNSChallenge/issues/20

I. How you calculated the power proxy?

For the number of synaptic operations per time-step (SynOPs), we utilized the following formula:

$$\text{SynOPs} = \sum_{\ell=1}^{L-1} \sum_{i=1}^{N^\ell} \gamma_i^\ell (N^{\ell+1} + N^\ell) \tag{5}$$

where γ_i^ℓ denotes the firing rate of neuron i in layer ℓ, N^ℓ represents the number of neurons in layer ℓ, and L is the total number of layers in the network. $\sum_{i=1}^{N^\ell} \gamma_i^\ell (N^{\ell+1} + N^\ell)$ signifies the summation over all neurons in layer ℓ. For each neuron, it multiplies the firing rate γ_i^ℓ by the sum of the number of neurons in the next layer $N^{\ell+1}$ (from outgoing connections) and the current layer N^ℓ (from recurrent connections). The overall SynOPs value is obtained by summing up the results from all layers in the network,

To calculate the number of neuron operations per time step (NeuronOPs), we summed up the number of neurons in each layer:

$$\text{NeuronOPs} = \sum_{\ell=1}^{L} N^\ell \tag{6}$$

J. 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	108.34	3.47	1798	7192

APPENDIX

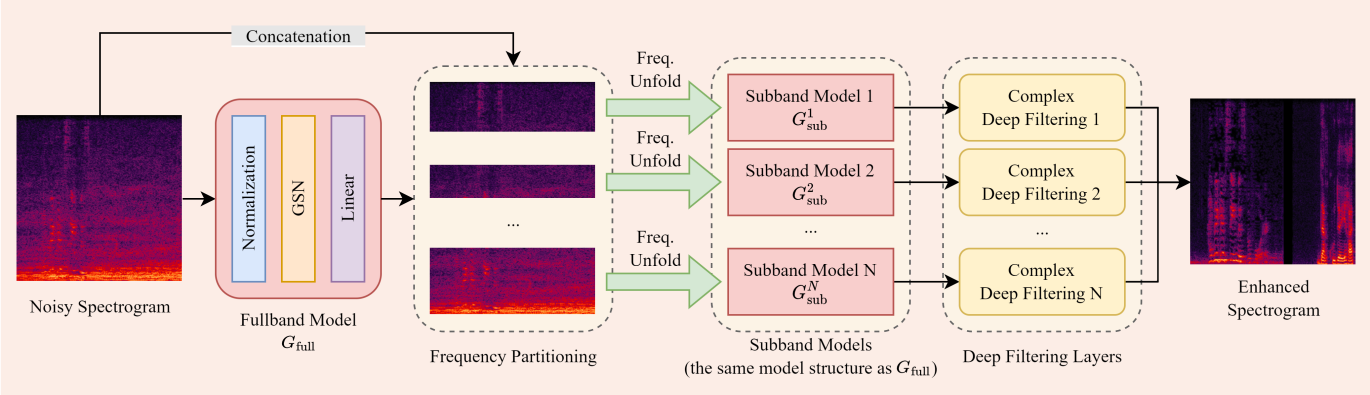

Fig. 1. Diagram of the proposed Spiking-FullSubNet architecture. It comprises a meticulously designed GSN as the core unit, encapsulated by an improved FullSubNet based on frequency partitioning.

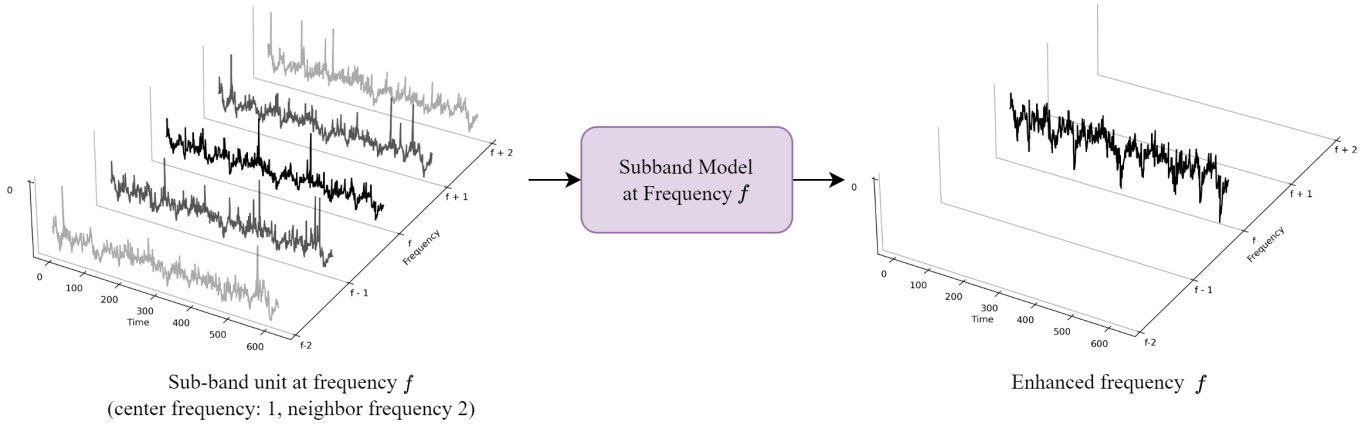

Fig. 2. Illustration of the internal processing of subband models within Spiking-FullSubNet. The subband models are operating in parallel, and each of them makes use of contextual information from its neighboring frequency bands.

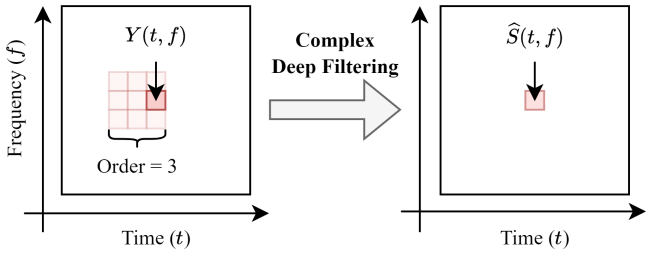

Fig. 3. Illustration of the complex deep filtering. Each time-frequency (T-F) bin is estimated using several T-F bins in its neighbors.

REFERENCES

[1] W. Fang, Z. Yu, Y. Chen, T. Masquelier, T. Huang, and Y. Tian, "Incorporating learnable membrane time constant to enhance learning of spiking neural networks," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 2661–2671.

[2] X. Hao, X. Su, R. Horaud, and X. Li, "Fullsubnet: A Full-Band and Sub-Band Fusion Model for Real-Time Single-Channel Speech Enhancement," in *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Jun. 2021, pp. 6633–6637, iSSN: 2379-190X.

[3] H. Schroter, A. N. Escalante-B, T. Rosenkranz, and A. Maier, "Deepfilternet: A Low Complexity Speech Enhancement Framework for Full-Band Audio Based On Deep Filtering," in *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, May 2022, pp. 7407–7411, iSSN: 2379-190X.

[4] S.-W. Fu, C.-F. Liao, Y. Tsao, and S.-D. Lin, "Metricgan: Generative adversarial networks based black-box metric scores optimization for speech enhancement," in *International Conference on Machine Learning (ICML)*, 2019.
[5] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz, J. Silovsky, G. Stemmer, and K. Vesely, "The kaldi speech recognition toolkit," in *IEEE 2011 Workshop on Automatic Speech Recognition and Understanding*. IEEE Signal Processing Society, Dec. 2011.
[6] S. Watanabe, T. Hori, S. Karita, T. Hayashi, J. Nishitoba, Y. Unno, N. Enrique Yalta Soplin, J. Heymann, M. Wiesner, N. Chen, A. Renduchintala, and T. Ochiai, "ESPnet: End-to-end speech processing toolkit," in *Proceedings of Interspeech*, 2018, pp. 2207–2211.
[7] Z.-Q. Wang, G. Wichern, S. Watanabe, and J. Le Roux, "STFT-Domain Neural Speech Enhancement With Very Low Algorithmic Latency," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 31, pp. 397–410, 2023. [Online]. Available: https://ieeexplore.ieee.org/document/9961936/
[8] J. Timcheck, S. B. Shrestha, D. B. D. Rubin, A. Kupryjanow, G. Orchard, L. Pindor, T. Shea, and M. Davies, "The Intel neuromorphic DNS challenge," *Neuromorphic Computing and Engineering*, vol. 3, no. 3, p. 034005, Aug. 2023, publisher: IOP Publishing.