Efficient Neuromorphic Data Processing: A Unified Framework for Pipeline Optimization and Lightweight Preprocessing*

*Note: Sub-titles are not captured in Xplore and should not be used

1st Given Name Surname
dept. name of organization (of Aff.)
name of organization (of Aff.)
City, Country
email address or ORCID

2nd Given Name Surname
dept. name of organization (of Aff.)
name of organization (of Aff.)
City, Country
email address or ORCID

3rd Given Name Surname
dept. name of organization (of Aff.)
name of organization (of Aff.)
City, Country
email address or ORCID

4th Given Name Surname
dept. name of organization (of Aff.)
name of organization (of Aff.)
City, Country
email address or ORCID

5th Given Name Surname
dept. name of organization (of Aff.)
name of organization (of Aff.)
City, Country
email address or ORCID

6th Given Name Surname
dept. name of organization (of Aff.)
name of organization (of Aff.)
City, Country
email address or ORCID

***Abstract*—To address the computational efficiency bottleneck in real-time processing of neuromorphic data (e.g., DVS128 Gesture), this paper proposes an end-to-end acceleration solution that integrates data pipeline optimization and lightweight preprocessing. At the system level, the use of `num_workers` for parallel loading and `prefetch_factor` for prefetching significantly enhances the throughput of the data pipeline. At the algorithmic level, a dynamic time window event frame aggregation method based on spatiotemporal sparsity is introduced, along with a low-computation pulse noise filtering module leveraging local spatiotemporal correlations. Additionally, structured model pruning is employed to reduce redundant connections and lower computational overhead. Experimental results demonstrate that on an NVIDIA 4070 GPU, the inference speed on the test set is doubled, while achieving classification accuracies of 92.36% (test set) and 98.97% (training set). This work combines data pipeline optimization with lightweight preprocessing, providing a high real-time solution for neuromorphic data processing in edge computing scenarios, with significant practical engineering value.**

***Index Terms*—Neuromorphic Computing, Data Pipeline Optimization, Pulse Noise Filtering**

I. Introduction

Neuromorphic data, characterized by its event-driven and sparsity-aware nature, has emerged as a promising paradigm for real-time applications such as gesture recognition, object tracking, and autonomous navigation. However, the efficient processing of such data remains a significant challenge, particularly in edge computing scenarios where computational resources are limited. A critical bottleneck lies in the substantial latency introduced by data loading and preprocessing, which can account for over 40% of the total training time. Traditional approaches to address this issue often focus on either algorithmic optimizations or hardware acceleration in isolation, failing to fully exploit the synergistic potential of a unified framework.

This paper proposes a novel solution that seamlessly integrates data pipeline optimization with lightweight preprocessing techniques to achieve end-to-end acceleration. At the system level, we leverage parallel loading and prefetching mechanisms to maximize data throughput. On the algorithmic front, we introduce a dynamic time window aggregation method that adaptively captures spatiotemporal sparsity in event data, alongside a computationally efficient pulse noise filtering module based on local spatiotemporal correlations. Additionally, we employ structured model pruning to reduce redundant connections, further enhancing computational efficiency.

Our contributions are threefold: 1) We present the first unified framework that combines data pipeline optimization with lightweight preprocessing, addressing the latency bottleneck in neuromorphic data processing. 2) We demonstrate significant improvements in inference speed (2× faster) and classification accuracy (92.36% on the test set) using the DVS128 Gesture dataset on an NVIDIA 4070 GPU. 3) We provide a comprehensive analysis of the trade-offs between speed, accuracy, and computational efficiency, offering practical insights for real-time applications.

This work bridges the gap between system-level optimizations and algorithmic innovations, paving the way for efficient neuromorphic data processing in resource-constrained environments. By reducing latency without compromising accuracy, our framework enables the deployment of event-based vision

Identify applicable funding agency here. If none, delete this.

systems in edge computing scenarios, such as drones, robotics, and IoT devices.

II. Related Work

A. Neuromorphic data preprocessing methods

Event compression and denoising

B. Deep learning training acceleration technology

Data loading optimization and model pruning

III. Methodology

IV. Experiments

A. Experimental Setup

Dataset Description In this experiment, we utilize the DVS Gesture dataset, a publicly available dataset for gesture recognition based on Dynamic Vision Sensor (DVS). The dataset comprises 1,342 samples, each corresponding to one of 11 distinct gesture categories. Each gesture is performed multiple times by approximately 29 different subjects. Each sample is stored in the form of an event stream. The dataset also provides training and test set splits, facilitating model evaluation on real-world event data. The data was collected under varying lighting and motion conditions to enhance diversity.

Evaluation Metrics To comprehensively evaluate model performance, we employ five key metrics: (1) Training speed (samples/sec), measuring computational efficiency by tracking processed samples per second during training; (2) Training accuracy (%), monitoring the model's learning progress by calculating correct predictions on training data; (3) Test accuracy (%), the primary classification metric evaluated on held-out test samples; (4) Training loss, recorded as an observational metric to analyze model behavior during training; (5) Test loss, evaluating generalization capability on held-out data.

All experiments were conducted on a single NVIDIA RTX 4070 GPU with a batch size of 16, using the Adam optimizer, Automatic Mixed Precision (AMP) for faster computation, and the CuPy library for GPU-accelerated array operations. This multi-faceted evaluation captures both computational efficiency (training speed) and model effectiveness (accuracy/loss), while the training loss serves purely as a diagnostic indicator rather than a control signal.

B. Impact of Frame Count on Model Performance

When processing DVS Gesture data, it is often necessary to convert the raw event stream into some form of image representation (e.g., event frames, accumulated frames, or time surfaces) so that it can be input into a convolutional neural network (CNN) for classification or other tasks. In this process, frame number is a key parameter, which indicates how many time windows the entire event stream should be divided into. Within each window, an "event frame" is generated as an input feature.

In this experiment, we process the DVS Gesture dataset on different frame numbers, as shown in Fig 1. Each row represents the processed dataset with different frame numbers,

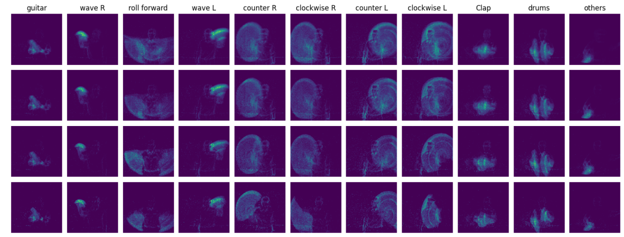
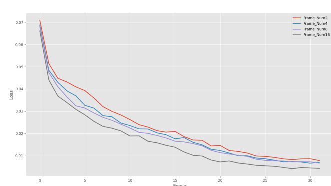


Fig. 1: DVS Gesture Dataset in Different Frame Numbers

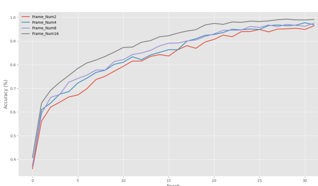
from top to bottom the number of frames is 2, 4, 8, and 16. And each column represents a different category, from left to right, which is guitar, right hand wave, forearm roll forward, left hand wave, right hand counter clockwise, right hand clockwise, left hand counter clockwise, left hand clockwise, clap, drums and others.

Through the sample of dataset, we can see that for most categories there is only a change in brightness caused by the difference of frame number, but for certain categories, such as the counter clockwise and clockwise rotation of the left and right hands, it is difficult to distinguish the corresponding images in the dataset with a small number of frames.

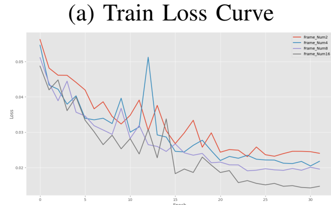
To detect the influence of frame number on the model training, we use the DVS Gesture dataset processed with different frame numbers to train the basic model and compare the performance of the model. The corresponding training results are shown in Fig2. The results show that the model with more



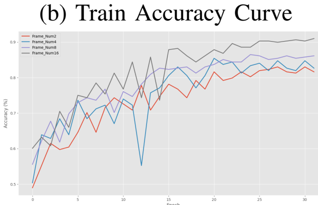
(a) Train Loss Curve



(b) Train Accuracy Curve



(c) Test Loss Curve



(d) Test Accuracy Curve

Fig. 2: Performance of Base Model at Different Frame Numbers

frames can achieve higher accuracy. The Fig2 demonstrate that the frame partitioning strategy exhibits negligible impact on both model loss and accuracy during the initial 10 training epochs. However, subsequent training reveals a significant performance divergence - higher frame counts consistently yield superior results, with the baseline model achieving lower loss values and higher accuracy when processing certain movement sampled frames. This phenomenon can be attributed to the increased information density afforded by finer tempo-

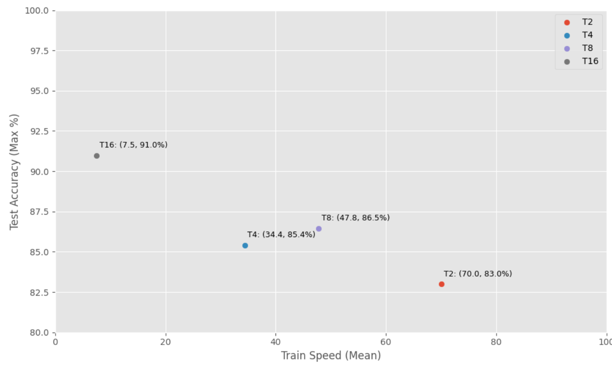


Fig. 3: Speed-Accuracy Plot

ral segmentation, which enables more comprehensive feature extraction throughout the network's deeper layers.

In addition to loss and accuracy, training speed is also an important metric for evaluating model performance. The Fig3 below illustrates the relationship between Average training speed and Maximum test accuracy during model training across different frame numbers.

As shown in Fig3, reducing the number of frames per event significantly accelerates the training speed. However, fewer frames lead to a certain degree of information loss, which consequently reduces the accuracy. In this experiment, the highest accuracy of 91.0% is achieved when each event is divided into 16 frames, while the training speed is the slowest at 7.5. Conversely, when each event is divided into 2 frames, the accuracy drops to its lowest at 83.0%, but the training speed increases nearly tenfold compared to the slowest speed.

C. Impact of Denoising on Model Efficacy

To further enhance the model's training performance, we performed denoising on the dataset based on setting the number of frames for event segmentation to 16, aiming to achieve superior training outcomes.We applied the spatiotemporal polarity consistency filtering method, as introduced in the Methodology chapter, to process the dataset. The frame

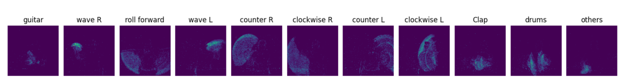
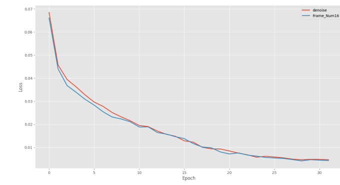


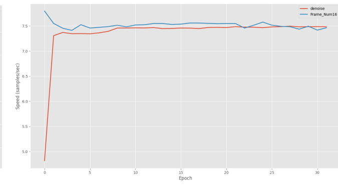
Fig. 4: Denoised Dataset

of each category from the processed dataset are illustrated in Fig7. The denoised images exhibit a high level of clarity and smoothness, with minimal visual artifacts. Essential details, such as edges and textures, are well-preserved, ensuring the integrity of the original data. The dataset presents a clean and visually consistent representation, characterized by a uniform and noise-free appearance. The overall visual quality of the denoised images is both natural and interpretable, making them suitable for subsequent analysis.

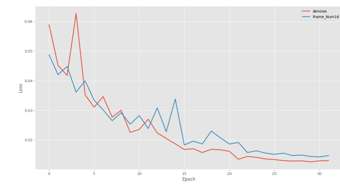
To further optimize based on the results of the previous experiment, we similarly segmented each event stream in the denoised dataset into 16 frames and trained the baseline model for 32 epochs to compare the training performance. The transformations of key parameters during the training process are shown in Fig5. By analyzing Fig5a and Fig5b,



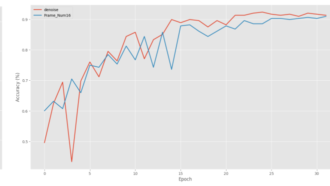
(a) Train Loss Curve



(b) Train Speed Curve



(c) Test Loss Curve

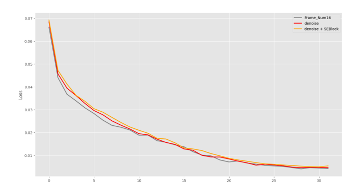


(d) Test Accuracy Curve

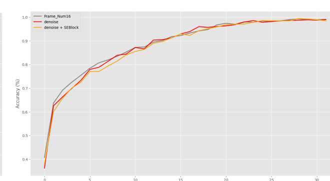
Fig. 5: Performance of Base Model at

which respectively depict the training loss and training speed of basic model on both the original dataset and the denoised dataset, it can be observed that except for the slower training speed of the base model in the first epoch on the denoised dataset compared to the original dataset, the performance of the two datasets during the training process is similar in all other cases. However, in the test process,

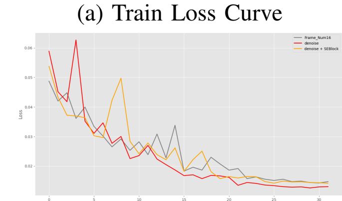
D. Assessing Model Performance with Attention Mechanism



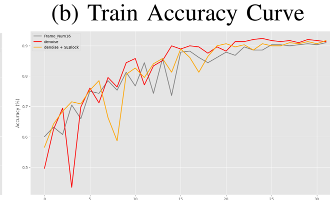
(a) Train Loss Curve



(b) Train Accuracy Curve



(c) Test Loss Curve



(d) Test Accuracy Curve

Fig. 6: Performance of Base Model at Different Frame Numbers

V. Conclusion

References

References

[1] G. Eason, B. Noble, and I. N. Sneddon, "On certain integrals of Lipschitz-Hankel type involving products of Bessel functions," Phil. Trans. Roy. Soc. London, vol. A247, pp. 529–551, April 1955.

[2] J. Clerk Maxwell, A Treatise on Electricity and Magnetism, 3rd ed., vol. 2. Oxford: Clarendon, 1892, pp.68–73.

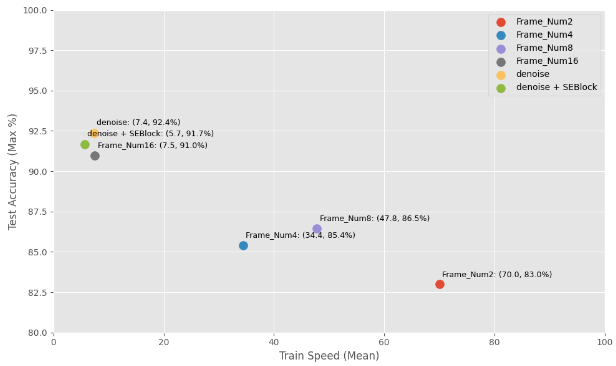


Fig. 7: Speed-Accuracy Plot of All Training Processes

[3] I. S. Jacobs and C. P. Bean, "Fine particles, thin films and exchange anisotropy," in Magnetism, vol. III, G. T. Rado and H. Suhl, Eds. New York: Academic, 1963, pp. 271–350.

[4] K. Elissa, "Title of paper if known," unpublished.

[5] R. Nicole, "Title of paper with only first word capitalized," J. Name Stand. Abbrev., in press.

[6] Y. Yorozu, M. Hirano, K. Oka, and Y. Tagawa, "Electron spectroscopy studies on magneto-optical media and plastic substrate interface," IEEE Transl. J. Magn. Japan, vol. 2, pp. 740–741, August 1987 [Digests 9th Annual Conf. Magnetics Japan, p. 301, 1982].

[7] M. Young, The Technical Writer's Handbook. Mill Valley, CA: University Science, 1989.