Event-Driven Letters: A Convolutional Spiking Neural Network for ASL Alphabet Recognition

Moretto Gabriele, Papini Francesco, Ruggeri Matteo, Tiberi Giulia
Politecnico di Torino

***Abstract*—This study presents a convolutional spiking neural network (CSNN) architecture developed to recognize the American Sign Language (ASL) alphabet. The network was trained and evaluated in software using Sinabs, a PyTorch-based library developed by SynSense. The study focuses on the preprocessing pipeline, the network architecture, and the classification performance obtained on the ASL-DVS dataset.**

I. Introduction

In recent years, deep learning applications have spread across all scientific domains, driving significant interest and advancement in this technology. However, a critical limitation arises from the inference energy consumption as neural network models scale, which reduces their sustainability for edge applications. To address this, spiking neural networks (SNNs) have emerged as one of the most promising paradigms. SNNs are bio-inspired neural networks that implement all communication via discrete spike events, rather than continuous real numbers, shifting the computing paradigm from clock-driven to event-driven. This has significant advantages on energy consumption, since now neurons will activate only when a spike reaches them, rather than at every clock cycle, like they do in normal ANN, making SNNs uniquely suited for energy-constrained environments.

However, fully exploiting the advantages of SNNs requires processing methods that preserve the sparsity and temporal structure of event streams. This motivates the study of event-based datasets and models specifically designed to handle asynchronous visual information.

A particularly compelling use case for event-based vision is gesture recognition, most notably American Sign Language (ASL) recognition, which is the primary communication medium for the deaf community. While markerless, vision-based ASL recognition has gained popularity due to its unobtrusive nature, it remains challenging because of high interclass similarities between signs, large intraclass variations among different users, and frequent finger occlusions. These complexities make ASL recognition an ideal benchmark for evaluating the robustness of event-based models.

For these reasons, in this paper, we present an application for the classification of the American Sign Language (ASL) alphabet using the ASL-DVS dataset. The study focuses on the software implementation of the preprocessing pipeline and CSNN, and on evaluating its recognition performance.

The remainder of this paper is organized as follows: Section II provides the necessary background on SNNs and CSNNs, alongside an overview of related literature. Section III describes the proposed architecture, the dataset processing, and the training procedure. Section IV presents the experimental results, and finally, Section V concludes the paper.

II. Background and Related Work

In this section, we explore the theoretical aspects of SNN and CSNN architectures and discuss related work in the field.

A. Related Work

One of the first frameworks for event-based ASL recognition was introduced by Rivera-Acosta et al. [1], whose approach avoids spiking dynamics, relying on a contour-based hand gesture processed by a simple artificial neural network. This method reached almost 80% of accuracy, validating the feasibility of the task.

Instead, focusing on SNN-based event stream classification, Chen et al. [2] explored the application of STBP-trained SNNs for sign language recognition, reporting a peak accuracy of 77% across 15 gesture classes. By introducing two distinct datasets - the natively recorded *DVS_Sign* (DAVIS346) and the *v2e*-converted *DVS_Sign_v2e* - the authors evaluated their model, achieving up to 77% accuracy. Their comparative analysis revealed that the native *DVS_Sign* dataset was consistently more challenging to classify due to acquisition noise and information loss caused by cropping.

Similarly, Ria Patel et al. [3] explored the application of CSNNs to the task of hand-gesture classification using event-based vision data. Their work was conducted on the ASL-DVS dataset too, with an IniVation DAVIS240C neuromorphic vision sensor. The study demonstrated that spiking neural networks can effectively process neuromorphic sensory data while maintaining computational efficiency. Experimental results showed that the proposed approach achieved a classification accuracy of 81%.

More recently, Lísková et al. [4] proposed an end-to-end neuromorphic pipeline for ASL fingerspelling recognition, combining a spiking visual attention mechanism with a CSNN. Their system achieved 92.27% accuracy in simulation. This work closely aligns with our research objectives, particularly regarding the relationship between architectural compactness and classification accuracy.

Building upon these works, we propose a CSNN for ASL recognition, consisting of three convolutional blocks ($2 \rightarrow 8 \rightarrow 16 \rightarrow 32$ channels, with spatial reduction from 128×128) followed by a 15-class readout layer.

B. SNNs

Spiking Neural Networks (SNNs), represent the third generation of neural computation, and they model the temporal dynamics of biological brains to process discrete, event-driven information. Within these architectures, synaptic weights contribute to a neuron's membrane potential, which integrates inputs over time until crossing a specific threshold to fire a spike. Crucially, while this framework operates with massive inherent parallelism, only a minute fraction of neurons is active at any given instant. This highly sparse activation allows SNNs to handle immense computational workloads at an energy cost orders of magnitude lower than conventional architectures, paving the way for a highly sustainable, biologically efficient generation of artificial intelligence.

Specifically, this behavior is formalized by models, such as the Integrate-and-Fire neuron, that idealizes the biological cell by simplifying its response into an electrical circuit. To a first approximation, neuronal dynamics can be formalized as a summation process coupled with a threshold-triggered mechanism.

For efficient numerical simulation, neuronal dynamics are typically formulated in discrete time: the discrete time evolution of the membrane potential V_i for an IAF neuron can be governed by the following relation:

$$V_i(t) = V_i(t-1) + \sum_j W_{ij} X_j(t) - V_{th} S_i(t-1)$$

where W_{ij} represents the synaptic weight connecting pre and post synaptic neuron.

The structural firing mechanism is driven by a Heaviside step function, which generate an output spike whenever the threshold is exceeded:

$$S_i(t) = \Theta(V_i(t) - V_{th}) = \begin{cases} 1, & \text{if } V_i(t) \geq V_{th} \\ 0, & \text{otherwise} \end{cases}$$

C. CSNNs

Standard fully connected SNNs suffer from some limitations when processing high-dimensional neuromorphic inputs, so Convolutional Spiking Neural Networks overcome these constraints by integrating a specific structure that comprehend weight sharing, local feature extraction and translation invariance.

A CSNN natively maps asynchronous events generated by neuromorphic sensors, such as a Dynamic Vision Sensor (DVS), by scaling information across three primary functional layers: convolutional, pooling, and fully connected layers. Typically, earlier layers capture fundamental spatio-temporal features, while deeper layers progressively integrate these localized responses into complex shapes and abstract category representations.

To process the events, the sensor output is aggregated using a variable temporal batch window to achieve a desired discrete resolution. Formally, an input sequence is represented as a sequence of tensors $S(n)$ for time steps $n = 0, 1, \ldots, N$. At each step, $S(n) \in \{0,1\}^{C \times U \times V}$ forms a sparse, binary multi-channel frame where U and V correspond to the frame width and height, and the channels C encode localized DVS event polarities. At each time step, the input tensor x (derived from $S(n)$) is convolved with a 2D kernel K to generate the spiking feature map y. This spatial convolution is defined as:

$$y[i,j] = \sum_m \sum_n x(i+m, j+n) \cdot K(m,n)$$

where (i,j) and (m,n) denote the spatial coordinates of the output feature map and the convolutional kernel, respectively.

The mean firing rate R_c of each output neuron, associated with class c, is computed by averaging its spike activity across all N timesteps:

$$R_c = \frac{1}{N} \sum_{n=0}^{N-1} S_c^{\text{out}}(n)$$

where $S_c^{\text{out}}(n)$ denotes the spike emitted by the output neuron associated with class c at timestep n. The predicted ASL gesture is selected as the class whose output neuron exhibits the highest mean firing rate:

$$\hat{y} = \underset{c}{\text{argmax}}\, R_c$$

III. Materials and Methods

A. Dataset

The ASL-DVS dataset consists of recordings acquired from five different subjects. For each subject, Dynamic Vision Sensor (DVS) events were collected while performing the American Sign Language (ASL) alphabet gestures, excluding the letters *J* and *Z*, whose dynamic trajectories differ significantly from the remaining static signs. The recordings are originally provided in the `.aedat2` file format and are subsequently converted to the more recent `.aedat4` format using the DV software suite developed by iniVation. This conversion ensures compatibility with modern event-based vision processing frameworks and facilitates subsequent data preprocessing and analysis.

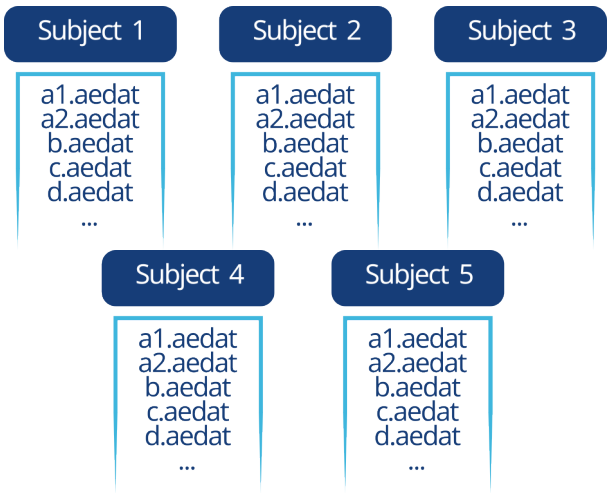


Fig. 1: Dataset structure

Initially, the dataset was partitioned by using *Subject 5* exclusively as the test set, while the recordings from the remaining four *subjects* were used for training. This subject-independent configuration was intended to evaluate the capability of the model to generalize to a subject that had not been observed during training. Under this setting, however, the classification accuracy did not exceed 40%.

A preliminary analysis of the recordings showed that *Subject 5* presents systematic differences from the remaining subjects. In particular, its recordings are considerably longer and therefore generate a larger number of samples when divided into fixed-duration temporal windows. As a consequence, using all the available windows without controlling their contribution could lead to an over-representation of this subject within the final dataset.

A qualitative inspection of the event frames also suggested differences in the spatial distribution and density of the recorded activity. Samples from *Subject 5* frequently exhibit a weaker and sparser event response, with a thinner hand silhouette and fewer clearly visible spatial details. In contrast, samples from the other subjects generally show denser event patterns, in which the structure of the hand and fingers is more clearly represented. These observations indicate that the variability between subjects affects not only the execution of the gestures, but also the temporal and spatial characteristics of the input signal.

However, the experimental results revealed that *Subject 5* samples were significantly different from those of the other subjects, leading to a substantial distribution mismatch between the training and test sets. The limited accuracy obtained with the initial partition was therefore interpreted as evidence of a considerable cross-subject distribution shift, rather than exclusively as a limitation of the proposed network architecture.

To obtain a more accurate results, a mixed-subject dataset split was adopted, where samples from all subjects were distributed between the training and test sets. This partitioning strategy was inspired by the dataset construction procedure adopted in [3], in which temporal samples from the different subjects are combined and shuffled before the creation of the final partitions. In this way, the different subject-specific acquisition characteristics, including those of *Subject 5*, are represented during training. This strategy provided a more balanced evaluation and enabled a fairer comparison of the different training configurations and preprocessing methods.

1) Preprocessing: To effectively extract spatiotemporal features from the raw, asynchronous ASL-DVS event streams, a dedicated preprocessing pipeline was implemented.

Since the original visual stream has a spatial resolution of 240×180 pixels, the coordinates are dynamically cropped to extract a localised 128×128 square window centered around the hand gesture, as shown in figure 2. To achieve this tracking, the raw spatial events within a designated time window are first collapsed into a static 2D density heatmap, which isolates the center of mass of the hand activity.

Furthermore, event-based cameras inherently generate background activity noise triggered by thermal fluctuations or minor lighting variations. To mitigate this and ensure high data sparsity, a spatiotemporal filter is applied directly to the cropped event stream, effectively isolating the actual gesture motion.

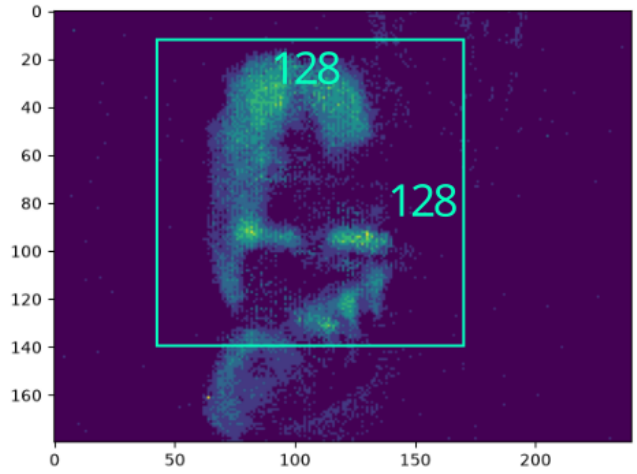


Fig. 2: Preprocessing method. The zone with the highest amount of spikes is cropped.

Finally, since Spiking Neural Networks (SNNs) process structural inputs over discrete time steps, the continuous and irregular event sequences are segmented into a sequence of fixed temporal windows. These accumulated segments are integrated into discrete frames according to a predetermined temporal resolution. To provide binary inputs to the spiking neurons, the accumulated values within each spatial frame are clipped to a strict binary state $S \in \{0, 1\}$, where 1 denotes a pixel with recorded event activity and 0 represents silence.

Another preprocessing strategy that was investigated consisted of extracting a fixed 180×180 region centered on the area exhibiting the highest event density, and subsequently resizing it to 128×128. The underlying motivation was to improve the spatial localization of the event activity by focusing on the most informative region of the sensor output, rather than processing the entire frame, which contains a larger number of sparse and potentially less relevant events.

However, this approach did not yield any improvement in classification accuracy compared to the previous preprocessing pipeline. A plausible explanation is that the cropping and subsequent resizing operations discarded relevant spatial information and introduced interpolation artifacts.

Another preprocessing strategy involved using shorter temporal integration windows (e.g., 10, 15, and $20\,\mathrm{ms}$), as suggested by Boyi Feng *et al.* [5], who reported promising results on a different but closely related event-based sign language dataset. The rationale behind this approach was that shorter time windows could better preserve the temporal dynamics of the event stream while reducing the overlap of unrelated events.

However, in our experiments, this strategy resulted in a significant degradation of the classification performance, since the optimal value that we found out for the ASL-DVS dataset is $50\,\mathrm{ms}$.

B. CSNNs

The preprocessed event frames are processed through a CSNN designed for software-based event-stream classification. The network processes input tensors of shape $B \times T \times C \times H \times W$, where B is the batch size, T represents the number of discrete timesteps, $C = 2$ corresponds to the ON/OFF event polarities, and H, W are the spatial dimensions.

The core architecture consists of three sequential spiking convolutional blocks designed for progressive feature extraction. Within each block, spatial feature extraction is achieved by applying a standard 2D convolution, which scales the feature map depth across the three layers from 2 input channels to 8, 16, and finally 32 output channels.

The continuous output of each convolution acts as the input current to a layer of Integrate-and-Fire (IAF) neurons. The IAF neuron model was chosen to represent the temporal integration of event-based inputs, with each layer configured with an independent, tunable threshold to optimize information flow.

Finally, to downsample spatial dimensionality while preserving the discrete event counts, a 2×2 sum pooling layer is applied, halving both the height and width in the output of each block, as illustrated in Figure 3. Following the convolutional blocks, the spatial dimensions are reduced to 16×16, and the tensor is flattened into an 8192-element vector per timestep. For the classification head, a fully connected linear layer maps these features to the 15 target ASL classes. Crucially, the final layer is passed through an IAF spiking activation rather than a standard continuous output, so that the network's predictions consist of discrete spikes over time.

1) CSNN Learning Method: To train the CSNN, the learning process relies on the Backpropagation Through Time, unrolling the network across the temporal dimension to evaluate sequential dependencies and compute parameter updates. Formally, this temporal dependency and error propagation can be formulated as follows [3]:

$$\frac{\partial \mathcal{L}}{\partial W} = \sum_t \frac{\partial \mathcal{L}(t)}{\partial W} = \sum_t \sum_{s \leq t} \frac{\partial \mathcal{L}(t)}{\partial S(s)} \frac{\partial S(s)}{\partial W}$$

where $S(s)$ represents the structural spiking activity at timestep s, illustrating how error signals flow backward through both spatial parameters and preceding recurrent temporal states.

However, because the derivative of the non-differentiable Heaviside step function is zero everywhere except where it is undefined, standard backpropagation would cause gradients to vanish, a challenge widely known as the "dead neuron problem." The network employs the Surrogate Gradient Method during training cycle. Specifically, during the backward pass a differentiable approximation of the step function is used to route the gradients. This approach ensures a stable and non-zero gradient flow, allowing the network to effectively update its convolutional and linear weights.

Crucially, because *IAFSqueeze* neurons retain membrane potentials across timesteps, their internal buffers must be explicitly managed to prevent unbounded gradient accumulation. So, at the end of each forward-backward cycle, the neuron states were systematically isolated from the computational graph: this implementation effectively operates as Truncated BPTT (TBPTT), ensuring full temporal isolation between consecutive independent ASL batches.

While the network's theoretical prediction is based on the cumulative spike count, the training phase utilizes the mean firing rate of each readout neuron across the temporal dimension to compute unnormalized logits. This averaging strategy allows for the application of a standard cross-entropy loss directly to the integrated outputs, stabilizing the gradient flow and effectively penalizing incorrect gesture classifications.

2) Hyperparameter optimization: To balance classification accuracy, structural spiking activity, and gradient stability, the network's optimal hyperparameters are systematically explored using the `Optuna` framework. Throughout this optimization study, core structural configurations are held constant, fixing the batch size at 5 and the convolutional kernel dimensions at 3×3. Because Spiking Neural Networks (SNNs) are highly sensitive to membrane potential dynamics, the optimization process focuses heavily on identifying four independent spiking thresholds (V_{th}) tailored to each specific layer, searching across a continuous range between 0.5 and 2.0. Concurrently, to maximize validation accuracy and prevent overfitting, the learning rate and weight decay are varied log-uniformly within the ranges of 10^{-5} to 10^{-3} and 10^{-6} to 10^{-3}, respectively. During each individual trial, the network is trained on the preprocessed subset, and its performance is evaluated. The specific configuration that yields the highest validation accuracy is automatically logged, establishing the optimal combination of layer-wise thresholds, learning rate, and regularization hyper-parameters utilized for the final execution phase.

IV. Results and discussion

The proposed model was subsequently trained and evaluated using the previously identified optimal hyperparameters (Table I).

TABLE I: Optimal Hyperparameters Identified via Optuna Optimization

Hyperparameter	Description	Optimized Value
`lr`	Learning Rate	7.39×10^{-5}
`wd_type`	Weight Decay Type	None
`spike_thresh_conv1`	Sp. Th. Conv Layer 1	1.10
`spike_thresh_conv2`	Sp. Th. Conv Layer 2	0.50
`spike_thresh_conv3`	Sp. Th. Conv Layer 3	0.81
`spike_thresh_fc`	Sp. Th. Fully Connected	1.02

Executed over a horizon of 40 epochs, the training process demonstrated that the CSNN could fully capture the spatiotemporal dynamics of the event-based dataset; the training loss curve decreased rapidly and stabilized near zero (Figure 4), while the corresponding training accuracy reached nearly 100%. Concurrently (Figure 5), the model achieved a maximum validation accuracy of approximately 82%, with the validation loss reaching a stable plateau around 1.0 after the first 15 epochs.

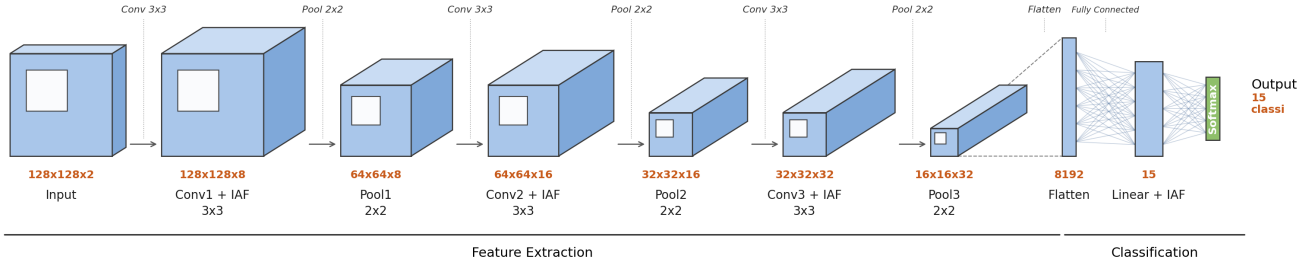


Fig. 3: The overall architecture of our CNN model. 'Conv.' and 'Pool.' denote the operations of convolution and pooling, respectively.

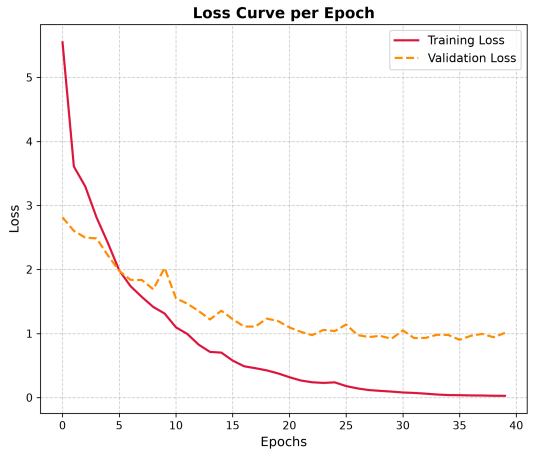


Fig. 4: Loss curves of training and validation process for 40 epochs

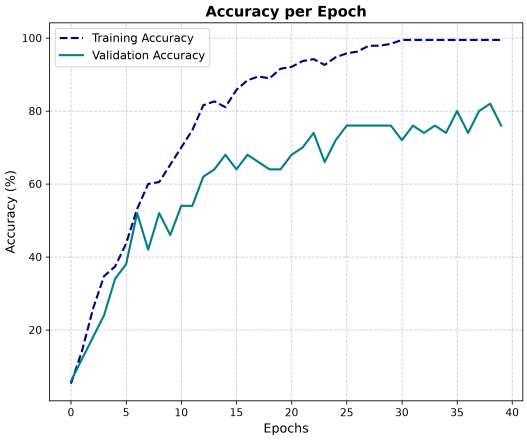


Fig. 5: Accuracy curves of training and validation process for 40 epochs

Then, a confusion matrix was computed exclusively on the test set to thoroughly analyze the predictive capabilities of the optimized CSNN (Figure 6). As expected, the matrix demonstrates a clear diagonal dominance, confirming that the vast majority of alphabet classes were correctly identified. Notably, several distinct gestures, such as 'a' and 'g', achieved perfect classification scores. This indicates that the combination of the dynamic cropping pipeline and the 50 ms temporal window is highly effective at tracking and isolating clear hand silhouettes.

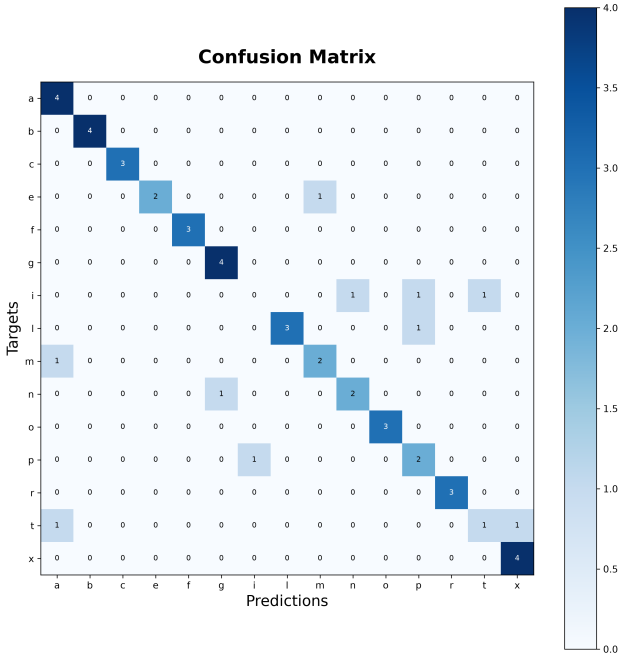
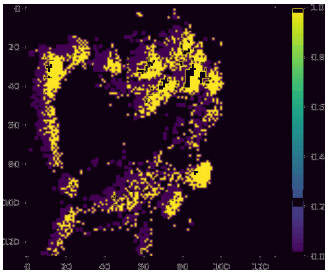


Fig. 6: Confusion matrix

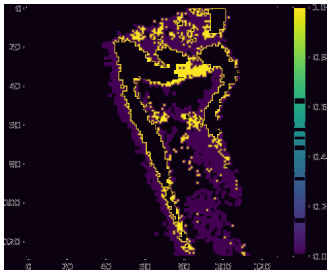
However, the matrix also highlights localized instances of inter-class confusion, which are largely intrinsic to the physical similarity of certain ASL signs (see Figure 7). For instance, the gesture 'm' (Fig:7c) is occasionally misclassified as 'e' (Fig:7b), while 'a' (Fig:7a) suffers from slight confusion with 'm'.

This confusion is likely due to the visual similarities between these signs, as illustrated in Figure 7; since the hand shapes and thumb positions for these letters are quite alike, the

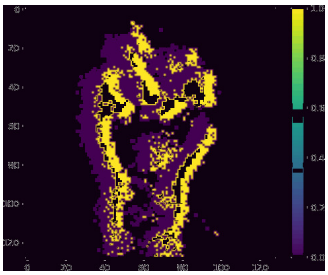
model faces difficulty in distinguishing between them, which is a common challenge in event-based gesture recognition.



(a) Letter A



(b) Letter E



(c) Letter M

Fig. 7: Visual similarity between the letters A, E, and M

This performance discrepancy can be attributed to the fundamental operational profile of the DVS camera. Because the neuromorphic sensor does not record absolute pixel intensity but instead captures asynchronous events triggered exclusively by temporal changes in log-intensity (i.e., motion), any static or slow-moving spatial overlaps during the execution of a gesture fail to generate sufficient spikes. Consequently, when complex hand configurations involve significant finger overlapping or occlusion, the resulting spatiotemporal features are not perfectly mapped by the camera, leading to a degradation in validation accuracy compared to the idealized training conditions.

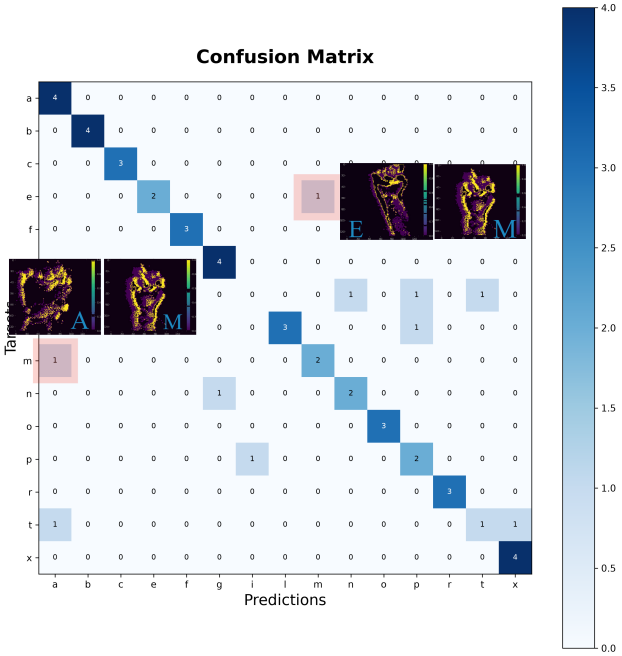


Fig. 8: Confusion matrix with confusing pairs

V. Conclusions

This study presented a Convolutional Spiking Neural Network for ASL recognition. The proposed pipeline combines dynamic RoI extraction and noise filtering with a CSNN trained on event-based data.

Our results confirm that SNNs can effectively extract complex spatiotemporal features from event-based data. After careful parameter tuning, the proposed model achieved a validation accuracy of 82% in software evaluation.

References

[1] M. Rivera-Acosta, S. Ortega-Cisneros, J. Rivera, and F. Sandoval-Ibarra, "American sign language alphabet recognition using a neuromorphic sensor and an artificial neural network," *Sensors*, vol. 17, no. 10, p. 2176, 2017.

[2] X. Chen, L. Su, J. Zhao, K. Qiu, N. Jiang, and G. Zhai, "Sign language gesture recognition and classification based on event camera with spiking neural networks," *Electronics*, vol. 12, no. 4, p. 786, 2023.

[3] R. Patel, S. Tripathy, Z. Sublett, S. An, and R. Patel, "Using csnns to perform event-based data processing classification on asl-dvs," 2024. [Online]. Available: https://arxiv.org/abs/2408.00611

[4] Š. Lísková, O. Vedmedenko, M. Fatahi, M. Hoffmann, P. M. Furlong, and G. D'Angelo, "Neuromorphic visual attention for sign-language recognition on spinnaker," *arXiv preprint arXiv:2605.06005*, 2026.

[5] B. Feng, R. Zhu, Y. Zhu, Y. Jin, and J. Ju, "Dynamic vision sensor-driven spiking neural networks for low-power event-based tracking and recognition," *Sensors*, vol. 25, no. 19, 2025. [Online]. Available: https://www.mdpi.com/1424-8220/25/19/6048