Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 235 (2024) 2674–2683

Procedia Computer Science

www.elsevier.com/locate/procedia

International Conference on Machine Learning and Data Engineering (ICMLDE 2023)

Sign Language Recognition using Spiking Neural Networks

Pranav Chaudhari[a], Alex Vicente-Sola[b,*], Amlan Basu[b], Davide L. Manna[b], Paul Kirkland[b], Gaetano Di Caterina[b]

[a] *Department of Information and Communication Technology, Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal, 576104, India*

[b] *Neuromorphic Sensor Signal Processing Lab, Centre for Image and Signal Processing, Electrical and Electronic Engineering Department, University of Strathclyde, Glasgow, UK*

Abstract

In recent years, research in automatic Sign Language Recognition (SLR) has undergone significant progress, serving as a foundational base for developing applications that aim to promote the integration of deaf individuals into society. Most of this progress is owed to the recent developments in deep learning. However, the deployment of conventional Artificial Neural Networks (ANNs) can be hindered by their requirements in terms of computational power and energy consumption. Therefore, to improve the efficiency of current SLR systems, in this work, we propose the use of the increasingly popular Spiking Neural Networks (SNNs), which, on the one hand, provide more energy-efficient computations than conventional ANNs and, on the other hand, are able to process temporal sequences with simpler architectures thanks to their temporal dynamics. To evaluate our method, we utilize WLASL300, the 300-word (300 classes of signs) dataset from Word-Level American Sign Language, and achieve an improvement in accuracy with the SNN (+2.70%) over the previous state-of-the-art, when working with energy-efficient spiking neurons. Furthermore, we construct a non-spiking version of the same network and evaluate it in a similar manner. Our results demonstrate how the SNN has sparser activations (25% less), thanks to the use of spiking neurons, and therefore can be implemented with a lower power requirement than an ANN version of the same architecture. This work thus demonstrates the possibility of performing SLR in a very effective and efficient way, thus opening up the development of applications that span from the automatic real-time translation of dynamic signs to remote control utilizing sign languages.

Peer-review under responsibility of the scientific committee of the International Conference on Machine Learning and Data Engineering

Keywords: Spiking Neural Networks; American Sign Language; Sign Language Recognition; Deep Learning

* Corresponding author.
E-mail address: alex.vicente-sola@strath.ac.uk

Peer-review under responsibility of the scientific committee of the International Conference on Machine Learning and Data Engineering
10.1016/j.procs.2024.04.252

1. Introduction

Sign language is a method of communication that uses hand gestures and motions, body language, and facial expressions rather than spoken words. Sign languages are a fundamental tool to allow communication between hearing people and deaf/hard-of-hearing communities. However, on the one hand, learning these signs for persons with hearing loss later in life is challenging [12]. On the other hand, people with hearing impairments from birth find it equally challenging to learn and use spoken languages. Nonetheless, the necessity for communication between people with and without hearing impairments prompts an investigation into automatic Sign Language Recognition (SLR) systems, to foster much-needed communication between those two communities [3].

Sign gestures can take the form of either static gestures or dynamic motions [5]. In the case of static gestures, identifying a sign as a specific word requires the processing of a single image frame. When dealing with dynamic gestures instead, a series of frames composing a short video is required to be processed to determine the word they refer to. This poses a much harder challenge to solve for any classifier given the spatio-temporal nature of the task. Nonetheless, sign languages employ several dynamic movements to represent words, hence the need to prioritize this subset of signs and focus on optimizing their correct and efficient classification. To this extent, we have used the Word-Level American Sign Language (WLASL) Dataset. This is one of the most extensive video datasets, comprising 11998 videos of different signs, which can be categorized into 2000 signs in total [8]. Fig. 1 shows sample frames of videos contained in the dataset.

In the history of Deep Learning (DL), feed-forward neural networks such as Convolutional Neural Networks (CNNs) have consistently achieved state-of-the-art results when handling non-temporal problems. However, specialized structures such as recurrent connections must be constructed to enable spatio-temporal feature extraction when the task involves understanding temporal and spatial information at the same time [21]. Spiking Neural Networks (SNNs) are biologically-inspired neural networks that have received increasing interest thanks to their key characteristics, such as temporal-based information processing [21], low latency, and low power consumption [1]. SNNs have gained significant popularity in the field of deep learning due to their ability to establish an end-to-end neuromorphic pipeline, featuring sparse and asynchronous data flow [7]. Previous work [21] has demonstrated how the inherent recurrency of the spiking neurons can allow to overcome the need for structural recurrences in the neural network. Therefore, in this work, we exploit such properties and apply an SNN-based approach to the SLR problem.

To help overcome the possible scarcity of data and achieve higher levels of accuracy, we utilize an SNN model that has been previously trained on the DVS Gestures dataset [21], a neuromorphic dataset comprising 11 different types of gestures performed by people, and we leverage transfer learning. We used a pre-trained model from the human activity recognition domain, as this is arguably close to the SLR domain. On the one hand, such an approach has been proven to be effective in offsetting the need that DL models have for thousands of labelled data points [4]; on the other hand, DVS Gestures data samples are arguably affine to that of SLR. Hence weights learned by solving such a classification problem can prove beneficial in learning features that are proper to the sign languages. We train our model on a subset of the WLASL dataset comprising 300 different classes of signs and compare it with five other state-of-the-art models. Furthermore, we construct a non-spiking version of the same network and evaluate it in a similar manner. Our results demonstrate how both versions of the network perform better than the state-of-the-art, with the ANN achieving slightly higher accuracy than the SNN; nonetheless the spiking network has sparser activations and, therefore, can be implemented with a lower power requirement than an ANN version of the same architecture.

The analysis of our results offers interesting insights as to the potential applicability of SNNs to solve this task, and to open up to the possibility of implementing highly-accurate SLR systems on edge devices, where the requirement for low power consumption often makes it difficult to deploy conventional DL-based solutions.

The paper is organized as follows. Section 2 reviews some automatic SLR systems presented in the literature. Section 3 covers the proposed model and its methods for implementation using spiking neural networks. Section 4 reports on the results and provides a discussion of the experimental data and the parameter-sweeping outcomes. Finally, Section 5 outlines our conclusion and future work.

Fig. 1. Sample frames from the WLASL Dataset which differ only in orientations of hand signs.

2. Literature review

Hassan et al. [5] have employed the slow-fast networks in a two-route CNN Model, in which the slow route analyses the raw video at a slow frame rate, while the faster route processes frames at a higher rate. Lateral connections are employed to merge elements of the quicker pathway to the slower pathway at each level. This is essentially equivalent to an average pooling after each route. The slower approach represents the slowly evolving semantics, whereas the faster road catches the rapidly changing ones. Sharma et al. [15] have proposed the use of a 3D CNN model, which can detect models of the patterns in volumetric data like videos. It is used to recognize dynamic sign language and may work independently for signers and their environment. This study extends the concept to real-time applications. While pre-processing, they convert video sequences to sequences of grey-scale images and remove unwanted noise using median filtering. Then the frames are processed and converted back to the video sequence to be fed into the neural networks. This approach uses the receptive field for filtering the features in the video sequences or the input image, with ReLu activation and stride = 1. To solve the problem of overfitting, the authors employ dropout layers with a ratio of 0.50 to 0.25. Convolutional layers have also been added, showing prominent results in lesser time requirements.

Garcia et al. [2] take the video of a particular sign, as input, and then extract individual frames from the video and calculates probabilities for each, using a CNN. They group the frames based on the character index and then use a language model to output a likelihood model. However, they pre-process the datasets to ensure these are equivalent and have the same image sizes. Two models were used for A-e and A-k, excluding j because of the lack of variations in the dataset. In Masood et al. [10], as video sequences contain both temporal-spatial features and temporal features separately, the authors have implemented the spatial features using the Inception model, which is a CNN, whereas the temporal features have been explored using a Recurrent Neural Network (RNN) on 46 classes.

Kiran et al. [6] have used 3D motion capture instead of 2D hand motion video capture. It has benefits such as immunity to lighting variations, motion blur, and external occlusions. They use 22 markers of 6.4mm for the hand and 14mm sized markers for the head. The system captures data on the cartesian plane. Markers are purposely placed to capture meaningful information from the signs performed. Singh et al. [17] have converted the primary picture to monochrome, which keeps it immune to blur and lighting conditions. This image is then reduced to 64*64 pixels, making it easier to generate features [20]. Pandey et al. [12] use hybrid CNN-based Hidden Markov Models, achieving encouraging results; however, clustering and high computational demands remain limiting factors in deployment. Raut et al. [14] use data captured through a webcam; images are converted to grayscale, and then the Otsu method is used to perform segmentation of the hand in the images.

Vedak et al. [19] use a Microsoft Kinect in an attempt to recognise Vietnamese sign language. This work uses a multiclass Support Vector Machine classifier, to distinguish both dynamic and static gestures. The features extracted through the Microsoft Kinect are pre-processed and then used for the recognition. Pramada et al. [13] use Hidden Markov Models for sign recognition, due to their inherent time-varying nature. This work leverages a set of basic hand shapes that may be used to formulate all gestures; these basic hand shapes are then used as input to the HMM, as the combination of these basic hand shapes forms specific hand signs.

Table 1. Dataset Statistics. The Dataset Name and Classes columns show the different sub-datasets of WLASL and the number of classes in each, respectively. The videos column reports the average number of videos per class. In order to keep the overall size of the dataset from growing too much, the number of videos per class is reduced as the number of classes increases. Finally, the Signers column reports the number of unique persons performing signs in each sub-dataset.

Dataset Name	Classes	Videos	Signers
WLASL 100	100	20.4	97
WLASL 300	300	17.1	109
WLASL 1000	1000	13.2	116
WLASL 2000	2000	10.5	119

Negri et al. [18] present a novel dataset for sign language recognition based on an event-based representation of sign gestures. The paper presents descriptive statistics of the dataset, including the number of samples, sign vocabulary size, and the distribution of sign gestures, shedding light on the diversity and scope of the dataset. The comparison was made between the event-based sign language dataset with traditional frame-based sign language datasets, highlighting the advantages of event-based representations in capturing the spatio-temporal dynamics of sign gestures and potentially improving recognition accuracy.

The VGG-GRU [22] model utilizes the VGG16 architecture, which is pre-trained on ImageNet, to extract spatial features. These are subsequently fed into a stacked GRU (Gated Recurrent Unit) structure. The Pose-GRU model [8], on the other hand, uses OpenPose to extract 2D key points for the body and hands. These key points extracted from these joints are then concatenated and sent into a two-layer stacked GRU. Pose-TGCN [11] models spatial and temporal relationships within the posture sequence, which is represented by body key points, using Temporal Graph Convolution Networks. Average pooling is performed along the temporal dimension, stacking several residual graph convolutional blocks to efficiently depicts the trajectory of the posture. The modified I3D model [16] adopts pre-trained weights from the I3D model and fine-tunes them on the Kinetics-400 dataset, with particular emphasis on hand shapes, orientations, and arm movements.

3. Methodology

3.1. Dataset

As mentioned previously, we seek to develop an efficient deep learning system that can accept dynamic sign language gestures as input, and classify them into the word they represent. Although different languages employ different sets of signs, without loss of generality, we focus on recognizing signs in American Sign Language (ASL). More specifically, we utilize the WLASL dataset, which comprises four sets of tasks with different classes and videos in each. Table 1 shows the statistics of the different classes and videos in the WLASL Dataset.

The WLASL dataset has 2000 words produced by 119 signers in different environments, making it the largest freely available ASL dataset, where its size allows to investigate word-level sign recognition within the context of our deep-learning study. The videos in the WLASL dataset naturally capture the temporal aspects and motion characteristics of sign language gestures, which are inherently dynamic and use continuous movements of the hands, arms, body, and facial expressions to convey meaning. The aim of our work is indeed to take into account the challenge of dynamic gestures, first converting the videos into individual image frames. As the videos in the dataset were recorded in different environments, to account for different lighting conditions and make them more uniform, we prepared the WLASL dataset by converting videos from .mp4 to .jpg format using the Video Streams to Frames codes [5], and then by transforming each frame into a grey-scale image. After this conversion, we apply data augmentation such as Random Crop and Random Horizontal Flip, at training time. The final images serve as input to our neural networks. We organized the data into a 1-column .csv file, containing video and image (final images after conversion) paths and their corresponding labels separated by spaces.

Table 2. Structure of the CNN architecture of the original ResNet used, showing number of layers and filters for each block. The n controls the depth of the network.

Output map size	32 * 32	16 * 16	8 * 8
no. of layers	1 + 2n	2n	2n
no. of filters	16	32	64

3.2. SNN and ANN architectures

The SNN employed in this work is based on the state-of-the-art architecture proposed in [20] [1]. The Spiking ResNet (S-ResNet) architecture consists of three convolutional blocks followed by downsampling (see Table 2 for the full architecture details). As defined in the original work, the network employs the LIF (Leaky Integrate-and-Fire) neuron model (Eq. 1). Despite being one of the simplest neuron models, the LIF is the most widely used model in the literature thanks to its simplicity and computational efficiency [9].

Let i represent a post-synaptic neuron, where $u_{i,t}$ represents its membrane potential, $o_{i,t}$ stands for its spiking activation, and λ denotes the leak factor (assigned a value of 0.874 following [20]). The index j refers to the pre-synaptic neuron, and the weights $w_{i,j}$ govern the synapse values connecting these neurons. The iterative update of the neuron activation is computed as follows:

$$o_{i,t} = g\left(\sum_j \left(w_{i,j} o_{j,t}\right) + \lambda \cdot u_{i,t-1}\right) \tag{1}$$

The function $g(x)$ (Eq. 2) acts as a thresholding mechanism, transforming voltage into spikes.

$$g(x) = \begin{cases} 1, & x \geq U_{th} \\ 0, & x < U_{th} \end{cases} \tag{2}$$

Following the spike, a reset occurs by subtracting U_{th} from the current membrane potential $u_{i,t}$, resulting in $u_{i,t}$ representing the membrane potential after resetting.

We employ a spikes-to-spikes (S2S) technique for the residual connection, and the output layer is created without leakage (λ) and without spiking activation ($U_{th} = \infty$). The final layer combines the output of the network over the entire sequence of time steps. Its voltage $u_{i,t}$ at the final time-step ($t = T$) produces the ultimate class scores (Eq. 3). Throughout the training process, these scores are contrasted against the actual ground truth labels utilizing cross-entropy loss. The network is then trained using Backpropagation Through Time (BPTT) alongside Stochastic Gradient Descent (SGD) and a momentum value set to 0.9 [21].

Furthermore, we also define a non-spiking version of the same architecture to enable a thorough comparison.

$$u_{i,T} = \sum_t^T \sum_j w_{i,j} o_{j,t} \tag{3}$$

[1] Code available at https://github.com/VicenteAlex/Spiking_ResNet

A triangular-shaped surrogate gradient is employed as its derivative to address the non-differentiability of the thresholding function. The parameter α is set to 0.3 for this purpose [21].

$$\frac{\partial o_{t,i}}{\partial u_{t,i}} = \alpha \; max\left\{0, 1 - \left\|u_{t,i}\right\|\right\} \tag{4}$$

Besides that, the S-ResNet [20] employs BNTT (Batch Normalization Through Time) as its normalization strategy. As shown in Eq. 5, the method establishes a separate BN module for each time step for a time-dependent input of d dimensions $x_t = (x_{1,t}...x_{d,t})$. This approach normalizes each feature k (or convolutional channel for CNNs) independently, similar to regular BN. However, it also defines individual statistics (mean $\mu_{k,t}$ and standard deviation $\sigma_{k,t}$) and learnable weights ($\gamma_{k,t}$ and $\beta_{k,t}$) for each time-step t [21].

$$BNTT\left(x_{k,t}\right) = \gamma_{k,t} \frac{x_{k,t} - \mu_{k,t}}{\left(\sigma_{k,t}\right)^2 + \epsilon} + \beta_{k,t} \tag{5}$$

In contrast, for non-spiking Artificial Neural Networks, instead of a neuron model, we employ the Rectified Linear Unit (ReLU) activation function and replace standard Batch Normalisation with Batch Normalisation via time. With these modifications, the network becomes a traditional feed-forward Residual Neural Network. These networks analyze the input instantly without regard for temporal dynamics, which implies that for a sequence classification problem like action recognition, they can provide an output per timestep, but not a global one for the whole sequence. We address this as done in the literature [21], by using the same output layer as the SNN, which can be considered as a voting system that combines the outputs for all timesteps by adding them together.

3.3. Training

Through supervised learning, we train our network to perform image classification. The final neuron layer is specified without a leak and cannot spike, to facilitate this classification. Consequently, the accumulated voltage in this layer after T timesteps is divided by T as the output value. The output class scores are then compared to the ground truth using a cross-entropy loss (Eq. 6), where C represents the number of classes, $u_{i,T}$ denotes the voltage of neuron i after the last timestep and y_i represents the ground truth labels [20]. When dealing with frame-based datasets, we chose to encode by convolution following the study in [20] which consists of converting the intensity values by directly passing the raw image through the first convolutional layer of the network (i.e., using the raw image as input).

$$L = -\sum_{i}^{C} y_i log\left(\frac{e^{u_{i,T}}}{\sum_{c}^{j} e^{u_{j,T}}}\right) \tag{6}$$

The weight updates for the learning process are calculated through BPTT. The derivative of the loss function concerning the network weights is the sum of neurons in the output layer and the sum of neurons in the hidden layers, as the final voltage at each layer depends on the contributions from all previous timesteps [20]:

$$\frac{\partial L}{\partial w_{i,j}} = \sum_{t=1}^{T} \frac{\partial L}{\partial u_{t,i}} \frac{\partial u_{t,i}}{\partial p_{t,i}} \frac{\partial p_{t,i}}{\partial w_{i,j}} \tag{7}$$

Table 3. Comparison between state of the art models on WLASL-300 Dataset

Models	Top-1 Accuracy (t)
Pose-GRU [8]	33.68
Pose-TGCN [11]	38.32
VGG-GRU [22]	19.31
I3D [16]	56.14
SlowFast [5]	79.63
Spiking Neural Networks (ours)	**82.39**
Non-Spiking Neural Network (ours)	**85.26**

where $p_{i,t}$ represents the current transmitted through the synapses after applying the weights [20].

$$p_{i,t} = \sum_j w_{i,j} o_{j,t} \tag{8}$$

Considering the temporal dependency of the membrane potential and its reliance on input spikes, the following equation is derived [20]:

$$\frac{\partial L}{\partial u_{t,i}} = \frac{\partial L}{\partial o_{t,i}} \frac{\partial o_{t,i}}{\partial u_{t,i}} + \frac{\partial L}{\partial u_{t+1,i}} \frac{\partial u_{t+1,i}}{\partial u_{t,i}} \tag{9}$$

Regarding the training hyperparameters, we use a learning rate schedule where the learning rate undergoes three divisions: at 57%, 78%, and 92% of the training process. Apart from that, the leak factor is set at 0.874, and the learning rate is set at 0.0268 with a batch size of 16 (following [20]). These are used consistently across all networks and datasets. The training was performed for a total of 65 epochs on an Nvidia GeForce GTX 1080 Ti GPU. The experiments amounted to approximately 72 hours of training in total.

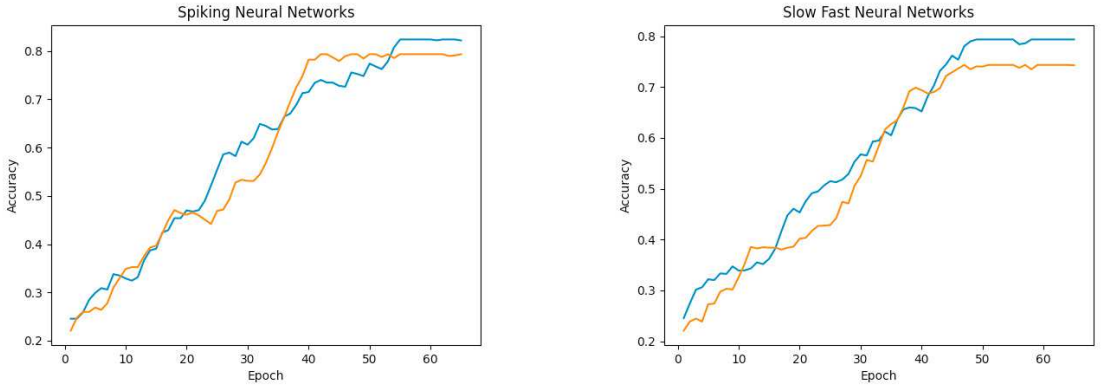
4. Results and discussion

4.1. Comparison with state-of-the-art models

We perform a thorough evaluation of recognition accuracy, comparing the proposed model with five other models, from literature, tested on the WLASL dataset (Table 3). The five models under consideration include VGG-GRU [22], Pose-GRU [8], Pose-TGCN [11], a modified I3D [16], and the SlowFast Model [5].

The evaluation centers on how the proposed model and the four other models perform on the WLASL datasets. Even though I3D is a larger model than the TGCN, Pose-TGCN can indeed achieve comparable results with I3D in terms of top-5 and top-10 accuracy, particularly on the large-scale subset WLASL300. This outcome serves as evidence that the TGCN efficiently encodes human motion information. However, it is essential to acknowledge that using an off-the-shelf pose estimator may lead to erroneous pose estimations, potentially affecting the overall recognition performance.

We re-trained I3D and SlowFast on WLASL 300 (as these networks were originally evaluated on different datasets) and compared top-1 accuracy scores. Early experiments showed how the I3D model struggled to converge using SGD, whereas Adam allowed the models to converge more reliably. Therefore, all the models were trained using Adam as an optimizer for a total of 65 epochs, which is where, on average, the validation accuracy saturated. Our results show that the proposed SNN can outperform all the other approaches and it achieves an improvement of nearly 3% accuracy points with respect to the previous state-of-the-art.



(a) Spiking Neural Networks (b) SlowFast Neural Networks

Fig. 2. Comparison of Our model with the latest state-of-the-art model SlowFast Neural Networks. It shows that our model has better accuracy than the SlowFast Neural Network. The blue line in both graphs shows the training accuracy while the orange line show the validation Accuracy.

In our experiments, we compared the top 1 error of our model against five other models. To this aim, we re-trained these five models on the WLASL 300 dataset and compare the top-1 accuracy of each. All models were trained using the Adam optimizer. Initially, we tried to train the I3D model with SGD, but we encountered convergence issues during fine-tuning with our datasets. Therefore, we switched to using the Adam optimizer for fine-tuning all models, and each model was trained for 65 epochs per subset until the validation accuracy stopped improving. Our experimental findings demonstrated that our model achieved a state-of-the-art level of accuracy on the WLASL 300 dataset, comprising 300 words or classes. It outperformed the recent state-of-the-art model by 2.70% in terms of top-1 accuracy (Fig. 2).

4.2. Energy efficiency and accuracy

Further to the comparison with other works, we build, train, and compare a non-spiking version of our SNN and record the activation percentage, which can be related to the power consumption that the two models would have if deployed on a physical SLR system. The activation percentage a was measured by extracting the activation of each neuron in the network $o_{i,t,n}$, and calculating their mean across all neurons i, all time-steps t per execution, and all the samples n in the test set:

$$a = \frac{\sum_{n=0}^{N} \sum_{t=0}^{T} \sum_{i=0}^{M} o_{i,t,n}}{NTM} \tag{10}$$

Results of this comparison can be found in Table 4. The number of spiking neurons that activated during the training phase sat around 71% as compared to the non-spiking neural network, which activates around 95% of neurons, hence making the latter network less sparse. Indeed, the spiking function in the SNN allows to only propagate the essential information, thus promoting a considerably higher degree of sparsity and increasing energy efficiency.

Notice that the original S-ResNet work [20] defines different residual connection implementations and demonstrates how some result in sparser networks than others. In this work we use the S2S implementation, which is the most accurate but also the least sparse. Moreover activation sparsity was not promoted in any way, as only cross-entropy loss was minimized. This suggests that this same network can be made much sparser if needed, while, at the same time, demonstrates that even without actively aiming at sparsity, just by using spiking neurons the network will have less neural activations.

Table 4. Comparison of the spiking version of our models versus the non-spiking version. Notably, the SNN achieves high levels of accuracy with almost 1/4 less power requirements.

Networks	Top-1 Accuracy (%)	Average Activation Percentage (%) (t)
Non-Spiking Neural Network (proposed)	**85.26**	94.9
Spiking Neural Networks (proposed)	82.39	**71.2**

Regarding accuracy, the SNN falls behind its non-spiking version (82.39% vs 85.26%), as expected. Despite the accuracy gap being relatively small, increasing SNN accuracy remains as a research direction of interest. Nonetheless, the energy efficiency of the SNN (-23.7 percentage points) is still advantageous in applications with memory and power consumption constraints.

5. Conclusion

In this work, we demonstrated the applicability of SNNs to the SLR problem and surpassed the accuracy of the previous state-of-the-art while using energy-efficient spiking neurons. This contributes to bypassing the limitations of deep learning in energy constraint systems, as SNNs can lead to energy efficiency gains when implemented in neuromorphic hardware, thanks to their sparsity. Moreover, the architecture employed consists of a single feed-forward network without recurrency, reducing the computational cost with respect to previous methods which use recurrent networks. As a result, this work opens the door for the development of more efficient applications of SLR.

Additionally, we demonstrate the difference in sparsity between the state-of-the-art S-ResNet SNN model and its non-spiking counterpart (ResNet), which serves as a functional example showcasing the efficiency gains of SNNs. As previously discussed, the implementation used for the spiking residual connection was the least sparse from the ones defined in [20], and the loss function did not enforce sparsity. The balance between sparsity and accuracy proposes a tradeoff where using fewer activations typically results in lower accuracy. Applications following this work can explore this tradeoff by switching to the other residual connection implementations or applying network regularization in order to make the SNN more sparse. On the contrary, when efficiency is not the goal, the proposed non-spiking network will be the optimal solution, as it provided slightly higher accuracy.

Overall, the positive results demonstrated in this work allow to prove the maturity of neuromorphic computing, providing accuracy levels that are comparable to those of non-spiking deep learning, while demonstrating their gains in network sparsity.

References

[1] Davies, M., Srinivasa, N., Lin, T., Chinya, G., Cao, Y., Choday, S.H., Dimou, G., Joshi, P., Imam, N., Jain, S., et al., 2018. Loihi: A neuromorphic manycore processor with on-chip learning. IEEE Micro 38, 82–99.

[2] Garcia, B., Viesca, S.A., 2016. Real-time American sign language recognition with convolutional neural networks. Convolutional Neural Networks for Visual Recognition 2, 8.

[3] Golekar, D., Bula, R., Hole, R., Katare, S., Parab, S., 2022. Sign language recognition using Python and OpenCV. Int Res J Modernization Eng Technol Sci 4, 1–5.

[4] Goodfellow, I., Bengio, Y., Courville, A., 2016. Deep learning. MIT press.

[5] Hassan, A., Elgabry, A., Hemayed, E., 2021. Enhanced dynamic sign language recognition using SlowFast Networks, in: 17th International Computer Engineering Conference (ICENCO), pp. 124–128.

[6] Kiran Kumar, E., Kishore, P.V.V., Sastry, A.S.C.S., Anil Kumar, D., 2018. 3D motion capture for Indian sign language recognition (SLR), in: Smart Computing and Informatics: Proceedings of the First International Conference on SCI 2016, Volume 2, Springer Singapore. pp. 21–29.

[7] Kirkland, P., Manna, D., Vicente-Sola, A., Di Caterina, G., 2022. Unsupervised spiking instance segmentation on event data using STDP features. IEEE Transactions on Computers 71, 2728–2739.

[8] Li, D., Rodriguez, C., Yu, X., Li, H., 2020. Word-level deep sign language recognition from video: A new large-scale dataset and methods comparison, in: IEEE/CVF Winter Conference on Applications of Computer Vision, pp. 1459–1469.

[9] Manna, D.L., Vicente-Sola, A., Kirkland, P., Bihl, T., Di Caterina, G., 2022. Simple and complex spiking neurons: Perspectives and analysis in a simple STDP scenario. Neuromorphic Computing and Engineering 2, 1–18.

[10] Masood, S., Srivastava, A., Thuwal, H.C., Ahmad, M., 2018. Real-Time Sign Language Gesture (Word) Recognition from Video Sequences Using CNN and RNN, in: Intelligent Engineering Informatics, Springer Singapore. pp. 623–632.

[11] Naz, N., Sajid, H., Ali, S., Hasan, O., Ehsan, M.K., 2023. Signgraph: An Efficient and Accurate Pose-Based Graph Convolution Approach Toward Sign Language Recognition. IEEE Access 11, 19135–19147.
[12] Pandey, P., Jain, V., 2015. Hand gesture recognition for sign language recognition: A review. International Journal of Science, Engineering and Technology Research (IJSETR) 4.
[13] Pramada, S., Saylee, D., Pranita, N., Samiksha, N., Vaidya, M., 2013. Intelligent sign language recognition using image processing. IOSR Journal of Engineering (IOSRJEN) 3, 45–51.
[14] Raut, M.D., Dhok, P., Machhale, K., Hora, J.M., 2015. A system for recognition of indian sign language for deaf people using otsu's algorithm. International Research Journal of Engineering and Technology (IRJET) 2.
[15] Sharma, S., Kumar, K., 2021. ASL-3DCNN: American sign language recognition technique using 3-D convolutional neural networks. Multimedia Tools and Applications 80, 26319–26331.
[16] Shi, B., Brentari, D., Shakhnarovich, G., Livescu, K., 2022. TTIC's WMT-SLT 22 Sign Language Translation System, in: Proceedings of the Seventh Conference on Machine Translation (WMT), Association for Computational Linguistics. pp. 989–993.
[17] Singh, G., Yadav, A.L., Sehgal, S.S., 2022. Sign language recognition using Python, in: 2022 International Conference on Cyber Resilience (ICCR), IEEE. pp. 1–5.
[18] Vasudevan, A., Negri, P., Linares-Barranco, B., Serrano-Gotarredona, T., 2020. Introduction and analysis of an event-based sign language dataset, in: 15th IEEE International Conference on Automatic Face and Gesture Recognition, pp. 675–682.
[19] Vedak, O., Zavre, P., Todkar, A., Patil, M., 2019. Sign language interpreter using image processing and machine learning. International Research Journal of Engineering and Technology (IRJET) 6, 1907–1909.
[20] Vicente-Sola, A., Manna, D.L., Kirkland, P., Di Caterina, G., Bihl, T., 2022. Keys to accurate feature extraction using residual spiking neural networks. Neuromorphic Computing and Engineering 2, 1–18.
[21] Vicente-Sola, A., Manna, D.L., Kirkland, P., Di Caterina, G., Bihl, T., 2023. Spiking Neural Networks for event-based action recognition: A new task to understand their advantage. arXiv preprint arXiv:2209.14915 .
[22] Walczynska, J., 2022. HandTalk: American sign language recognition by 3D-CNNs. Ph.D. thesis. University of Groningen.