Proceedings of the 2nd International Conference on Machine Learning and Automation
DOI: 10.54254/2755-2721/107/20241399

Enhancing Real-Time Vision Systems: Integrating Dynamic Vision Sensors with Vision Transformers to Increase Computational Efficiency

Annabelle Yao[1,4,*], Oliver Su[2,5], Sumedha Kumar[3,6]

[1]The Lawrenceville School, New Jersey, USA
[2]Alexander W. Dreyfoos School of the Arts, Florida, USA
[3]Monta Vista High School, California, USA

[4]ayhk2017@gmail.com
[5]oliver.su.2008@gmail.com
[6]sumedhakumar2@gmail.com
*corresponding author
Annabelle Yao is the first author, Oliver Su and Sumedha Kumar are co-second authors.

Abstract. Optimizing and understanding vision systems' computational accuracy is crucial as they become increasingly integrated into daily life. Dynamic Vision Sensors (DVS) and Vision Transformers (ViTs) lead computer vision technology with efficient object recognition and image processing. However, DVS data's high computational complexity poses a problem for its real-time implementations. Merging these technologies can enhance vision system performance in dynamic environments and optimize real-time DVS processing. In our work, we use a ViT architecture to classify the DVS 128 dataset and compare our results with existing works using SNNs. We analyze how our method affects accuracy and loss, experimenting with different DVS-to-ViT input patch sizes. Our results show that the large patch size of 32x32 pixels has better accuracy and smaller loss than the 4x4 pixel patches as epochs increase. Our method also achieved a high 98.4% accuracy and low 0.22 loss within five epochs, significantly outperforming previous works averaging 93.13% accuracy over more epochs. These results highlight ViTs' large potential for real-time DVS data classification in applications that require high accuracy, like autonomous vehicles and surveillance systems.

Keywords: Dynamic Vision Sensors, Vision Transformer, Deep Learning, Vision Systems, Gesture Recognition.

1. Introduction

The advancement of computer vision technologies has revolutionized and driven significant improvements in the field, whether in autonomous driving or medical imaging applications. Traditional vision systems relying on frame- based cameras have succeeded in processing image data. However, they face numerous limitations in moving environments, challenged with motion blur and high latency within their system.[1] In recent years, the emergence of Dynamic Vision Sensors (DVS), inspired by the biology of the human retina, has offered a possible solution to such problems. Yet, even as the usage

DOI: 10.54254/2755-2721/107/20241399

of DVS increases, real-time applications and data processing have proved to be a continued challenge due to the high computational complexity of DVS systems.

DVS sensors, also known as event-based cameras, detect changes in their current scene from their previously captured scene, capturing frames at a high temporal resolution in a fixed interval.[2] Each DVS pixel operates independently from the others, creating events whenever a change in local light intensity is detected.[3] This allows for capturing fast-moving objects without motion blur and reduces the amount of redundant and unused information stored in the system's memory. It also reduces the delay between sensing and processing information, which traditional vision sensors are subject to.

Similarly, Vision Transformers (ViT) are powerful tools in vision processing. In deep learning, their architecture is frequently used due to its high computational efficiency and self-attention mechanisms. They are a powerful machine-learning model that can handle large datasets and complex inputs.[4] ViT's transformer architecture also helps it weigh the importance of different input parts, capturing global dependencies and contextual relationships, which may aid in the processing of DVS gesture data.

In this work, we combine DVS's enhanced motion-capturing abilities with Vision Transformer's (ViT) high image computational efficiency, which can help improve the vision sensor's performance in dynamic, fast-changing environments in real time. Our research will explore how combining the two can enhance DVS gesture recognition data's processing efficiency and scalability, potentially reducing computational requirements and optimizing real-time DVS gesture classification. We also investigate how different-sized inputs may enhance or affect the accuracy and loss of the Vision Transformer results.

In sum, our research makes the following contributions:

- We identified a new method of implementing a Vision Transformer Architecture to process the DVS 128 Gesture dataset and analyzed its ability to perform classification
- We modeled and compared our method with previous experiments, analyzing the results and drawing conclusions
- We experimented with different-sized inputs of DVS data to the Vision Transformer and investigated the impact on accuracy and loss

Referencing previous works, we found information that will help improve our research. A study conducted at TCS Research & Innovation in Kolkata, India, revealed that vision-based solutions for identifying gestures and speech require heavy computational power and are highly inefficient.[5] As a result, a spiking neural network (SNN) incorporating convolution and reservoir computing was designed. In the work, the SNN first learns the spatial features of the gesture DVS dataset by performing convolution over input data, and then the temporal features are learned by using a reservoir layer of spiking neurons.[6] This paper achieved an overall accuracy of 89% over eight gesture classifications and observed that the network uses fewer parameters for learning. In another similar work that investigates SNNs[7], the paper observed a trend of a high percentage of accuracy when DVS cameras were input in SNNs. Our work will further these findings by researching the impact of DVS gesture data as input to another machine learning model type: transformers, specifically ViTs, in optimizing real-time dynamic visual data analysis.

The study confirms that dynamic vision sensing yields highly accurate results. Our study will attempt to determine if combining that high precision with a less power-intensive medium will result in a stronger overall system.

Another study conducted at the Institute of Neuroinformatics in Zurich, Switzerland, discovered that the DVS performed differently when certain parameters, such as bandwidth and sensitivity, were prioritized. This is known as optimizing biases. DVS systems, although exceptional at capturing critical, dynamic information, run on complex circuitry that hampers their efficiency.[8] Every bias has an impact on how the other biases can be utilized. There will always be a tradeoff between benefits and detriments. This will be considered when running our DVS data and writing our analysis, as it will note the possible impacts of hardware on the final results observed.

In our case, we are facilitating the analysis of the provided video data by introducing another medium. Currently, much of the vision sensor's potential is limited by its energy requirements. We hope to

DOI: 10.54254/2755-2721/107/20241399

allocate more energy to other critical parameters by increasing the threshold at which the sensor becomes inefficient. By making the overall system more formidable, we expect our results to remain highly accurate while decreasing computational power requirements. If successful, this will make DVS sensors more practical and accessible. We will attempt to find an optimal balance that maintains a high accuracy while not requiring large amounts of power to run.

A study conducted across several Shanghai institutes states that because transformers are highly sensitive to pixels and patches, they outperform many other systems.[9] The NLP Gumbel-Softmax transformer architecture was optimized for use with a decision network. The purpose of this is to pressure the program to reduce losses, which helps maintain accuracy while reducing the cost of computation. Energy is not wasted on lossy results. The study focuses on optimizing how data is fed to the network, which reduces the amount of power required to run the program efficiently. We will use data from a vision sensor and run it through a ViT, which will break down the information into patches. This process is expected to increase efficiency greatly.

We will compare our results with those from a 2021 study conducted by the Semiconductor Research Corporation (SRC). The project done at the SRC attempted to solve the problem of sparse DVS data limiting the potential of SNN analysis. Researchers proposed the Spike Activation Lift Training (SALT), which increases spike activity by optimizing weights across convolutional layers.[10] SALT utilizes sequential optimization, which calculates the spike activation of each consecutive layer. As a result of this technique, the experiment yielded a high increase in performance, with a final accuracy of around 85% across multiple time steps, as compared to around 75% accuracy without SALT. Our experiments will also attempt to increase performance while maintaining practical computational energy levels by analyzing smaller packets of information, courtesy of the ViT. By comparing our results at the end, it can be determined tentatively if our method of analyzing DVS data in real-time provides any profound benefits.

Convolutional neural networks, although practical, are not the most efficient method of analyzing video data. Vision transformers (ViT) have been considered a viable replacement for CNNs, as they are proficient at object detection and image comprehension.[11] Researchers have determined that both CNNs and transformers do their tasks with equal benefits and disadvantages. By manipulating certain factors, we aim to determine how integrating DVS with ViTs further optimizes the computational process compared to past research. Although CNNs and ViTs possess certain common features, there are distinct variances. While other neural networks are also sufficient for the task, ViTs are a notable candidate for visual analysis.[12] Our study will consider the efficiency of implementing a ViT structure in dynamic vision sensing.

2. Methods

We directly used an existing DVS dataset imported from the Tonic library[13] and preprocessed the data to ensure the data type fit the model's requirements before training. To train our machine learning algorithm, we implement the library Pytorch[14] to run the DVS data through our written ViT model. Then, we compare the results with existing research on optimizing Spiking Neural Networks (SNN) for DVS.[15] We also use a pre-existing dataset, the DVS128 Gesture Dataset from SpikingJelly, to classify hand gestures.[16] We plan to train our vision transformer on this video dataset.

2.1. Dynamic Vision Sensors

The DVS format is based on text files, where each line corresponds to a single event, and DVS data is commonly saved as a .npz file. The format consists of four pieces of information: timestamp, x-coordinate, y-coordinate, and polarity. The timestamp is when the event occurred and is essential to reconstructing temporal information and understanding the precise order of events. The X and Y coordinates represent the pixel's coordinates where the event occurred, recording changes in intensity at specific pixel locations. Finally, the polarity indicates whether the pixel's intensity increased or decreased at the given timestamp in relation to the previous information the DVS captured. After receiving information via light intensity, the sensor will convert the data into electrical signals. Then,

DOI: 10.54254/2755-2721/107/20241399

the information goes through a comparator, which interprets the signal as positive or negative. After processing, the information is recorded as an event.[17]

Figure 1 shows how the incidence light luminance curve causes changes in spiking. When the voltage curve goes below the negative threshold, or the horizontal blue lines, a negative event is output.[18] This means that the luminance in that pixel has dimmed. Conversely, when the curve passes through the higher threshold, indicated by the red line, a positive event is output; in this case, the luminance in the pixel has been detected to increase.

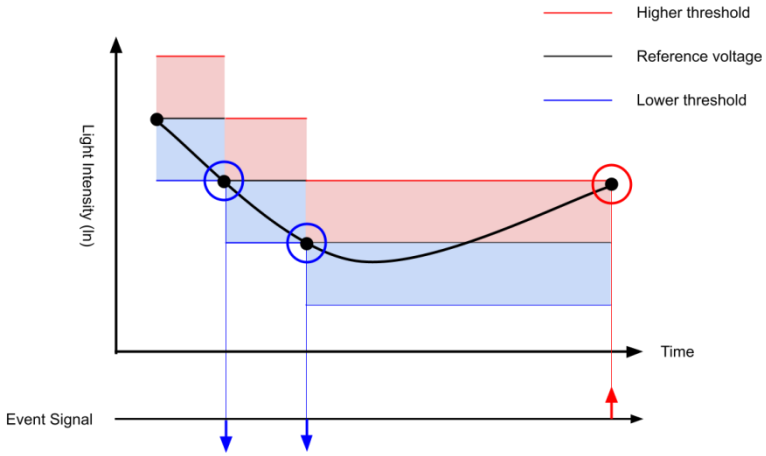

Figure 1. This chart displays the incidence light luminance going below the negative threshold and above the positive threshold as time continues.

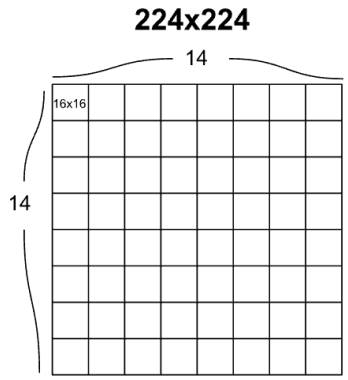

FIG. 2. Diagram of a frame split into 14, 16x16 pixel patches

Figure 2. Diagram of a frame split into 14, 16×16 pixel patches

Using this format of DVS data, we adjusted the input to the Vision Transformer by splitting the DVS video data into a set number of 16 frames, equally spaced throughout the time interval of the entire video. An example of the original data visualized in frames is below in Figure 3. Then, each frame was downsampled using the tonic library's transforms.downsample function into 14 by 14 grid of 16 x 16 pixel frames, shown in Figure 2. 3 example frames are visualised in Figure 4. These downsampled frames are then saved into .jpg files under the same folder (for frames from the same video) to be input later into the Vision Transformer.

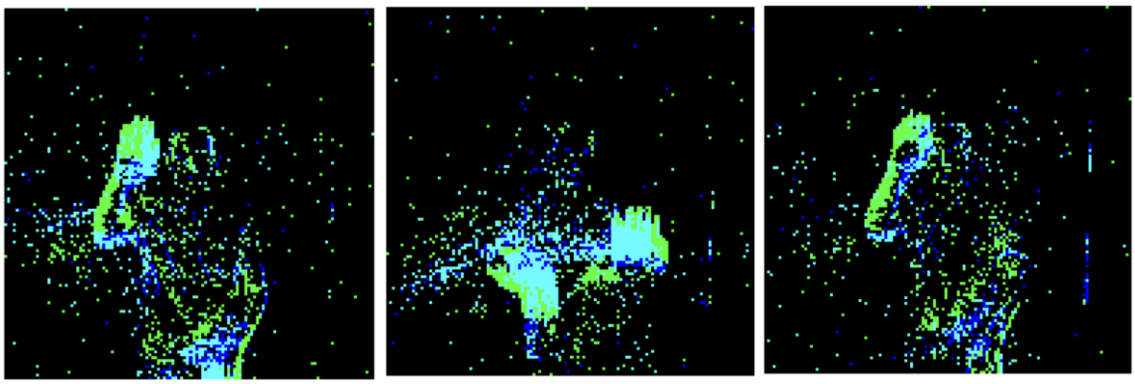

Figure 3. Examples of visualized DVS 128 Gesture data using the tonic library

DOI: 10.54254/2755-2721/107/20241399

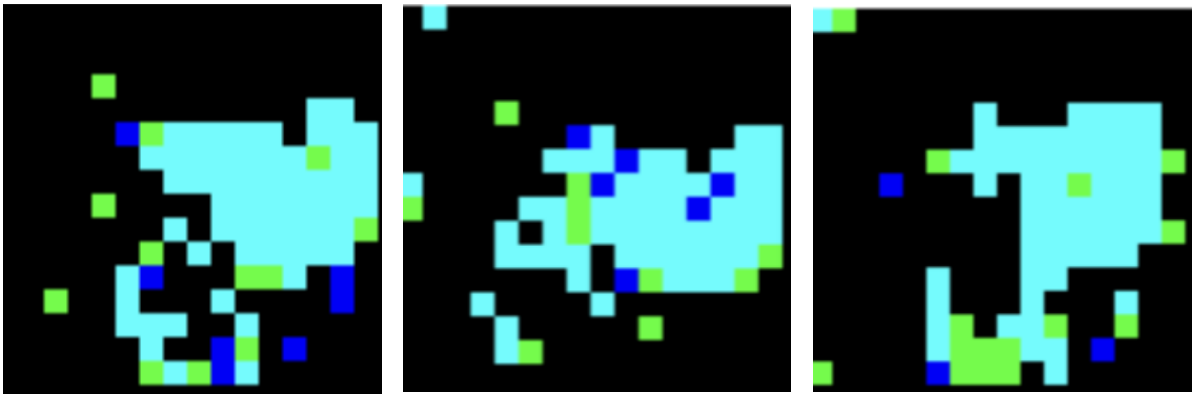

Figure 4. Examples of visualized DVS 128 Gesture dataset after downsampling into 16x16 pixelated frames

2.2. Vision Transformer

After data preprocessing and importing all the necessary libraries (PyTorch, NumPy, Matplotlib, Tonic), we build the model architecture, which defines its structure and determines how it will process the input data. Then, we train the model, where we feed it with input images and the corresponding labels and adjust parameters to minimize the loss function. Each step is explained in more detail below.

Upon running the ViT, the image will be converted into a series of patches and transformed into embeddings using a linear transformation to the flattened pixel values of the patch. These embeddings project data into vector space where similar data has similar vectors. Next, a classification token is added to the input sequence, which gathers information from all patches for classification. This token is learnable and helps create a representation of the image. Then, positional embeddings are added to each input to retain where the patches are, which regular Transformers tend to lack. Finally, the Transformer Encoder processes the sequence. This includes multi-head attention, layer normalization, feed-forward network, and residual connections. Multi-head attention allows information sharing between patches, Layer normalization stabilizes and speeds up training, Feed-forward networks process the output of the attention mechanism, and residual connections improve information flow and prevent vanishing gradients.

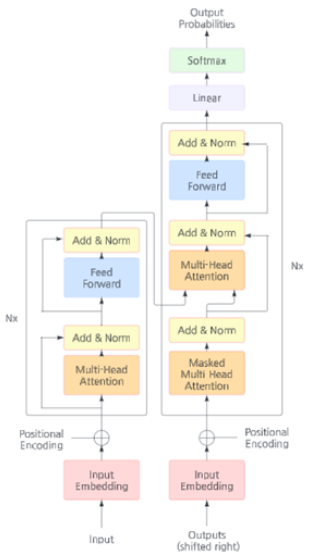

Figure 5. Model of the ViT architecture

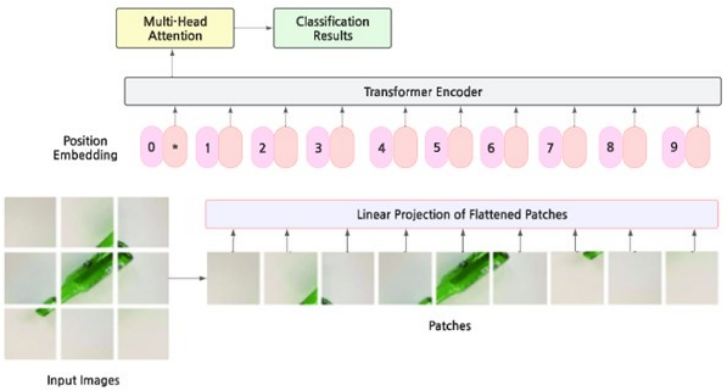

Figure 6. Overall structure of how ViT works, processing input/output

DOI: 10.54254/2755-2721/107/20241399

Fig. 4 shows a select sample of frames that display results from varying sensors with different parameters. These datasets are split into training and testing sets using the tonic.datasets.DVSGesture function with "train" being True for the training dataset and "train" being False for the test dataset. From there, with ViT, we will train our program to run through the training data, which will perform binary classification tasks with the different kinds of poses. The training loop involves loading the data in batches, computing Cross Entropy loss and accuracy, and updating the model parameters through backpropagation. The ViT was trained with a total of 20 epochs at a learning rate of 1e4 and a batch size of 32 (the same as the number of frames a single DVS video is split into). The exact code for the ViT is attached in our GitHub here.[19] Finally, we run our model through the testing dataset to classify the transformed DVS dataset.

For further investigation on how patch sizes taken from each frame may affect the overall accuracy and loss of ViT's DVS data classification, we go through the same process described above. Each time, we adjust the patch size so that it is either 4x4, 8x8, 16x16, or 32x32.

2.3. Metrics

To measure how well our model does against a baseline, we will use metrics like test accuracy, cross-entropy loss, and runtime to determine how quickly and accurately the gestures can be classified.[20] This metric will show how much our method optimizes time and memory. We also plan to look at average loss functions, specifically categorical cross-entropy. The categorical cross-entropy loss function measures the difference between the predicted probability distribution and the actual distribution by calculating the negative log-likelihood of the true class's predicted probability and then averaging this value over all data points. It indicates how well the model's predictions align with the true categories.[21] With categorical cross-entropy, there are many poses to classify, which is why this metric is more relevant than just calculating the Area under the ROC Curve (AUC ROC) for each category. If our program classifies the human pose in the correct category, it is a "correct" prediction. We aim to have the metrics (test accuracy, training time, and categorical cross-entropy) from our own trained model to be better and get more effective results than existing SNN models from past research.

3. Results

3.1. Experiment Results

Our experiment disproved our original hypothesis that a vision transformer would perform significantly better than other models on DVS input data. The specific data found during our research is presented in Table 1.

Table 1. Data tables of our Vision Transformer model's accuracy in classifying DVS gesture data as the number of epochs increases.

Patch Size 4×4 pixels		
Epoch	Loss	Accuracy
1	1.581	0.637
2	0.854	0.838
3	0.544	0.9
4	0.38	0.935
5	0.272	0.955

Patch Size 8×8 pixels		
Epoch	Loss	Accuracy
1	1.603	0.627
2	0.853	0.843
3	0.54	0.913
4	0.357	0.942
5	0.258	0.964

Patch Size 16×16 pixels		
Epoch	Loss	Accuracy
1	1.688	0.579
2	0.917	0.809
3	0.578	0.923
4	0.364	0.962
5	0.238	0.975

Patch Size 32×32 pixels		
Epoch	Loss	Accuracy
1	1.881	0.474
2	1.052	0.821
3	0.627	0.929
4	0.371	0.965
5	0.22	0.984

DOI: 10.54254/2755-2721/107/20241399

Our experiment analyzes how the accuracy and loss change based on how big the patches are. We test these metrics on four different sizes: 4x4 pixels, 8x8 pixels, 16x16 pixels, and 32x32 pixels; we observed that patch size makes a large impact on the behavior of these metrics. All of these patches were run through 5 epochs. In the same amount of epochs, the accuracy for each patch grew with the size of the patch. After five epochs, the accuracy peaks at 98.4% when using larger 32x32 pixel patches. The lowest accuracy was 95.5% with smaller 4x4 pixel patches.

Similarly, when analyzing the loss calculation of each patch size at five epochs, the loss was minimized most at 32x32 pixel patches, with a loss of 0.22. The loss was the largest at 4x4 pixels, at 0.272. We expect our model to perform best when the loss is at its lowest and the accuracy is at its highest. This shows that using the largest 32x32 patches is optimal for classification on our vision transformer, as the loss is at the lowest and the accuracy is at the highest. The more information the transformer analyzes, the more accurate it will be due to a more refined image.

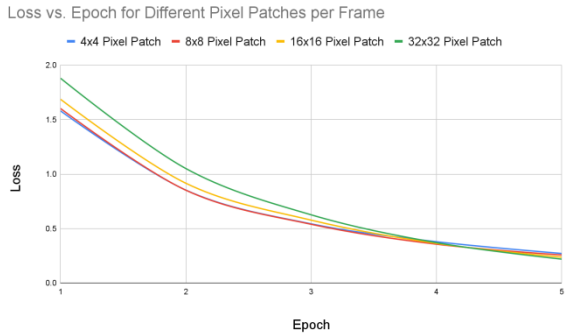

Figure 7. Analyzes patch size changes with loss

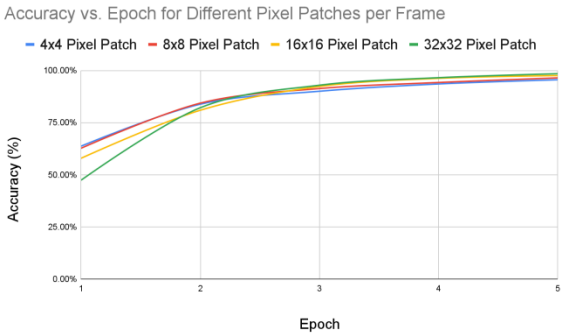

Figure 8. Analyzes patch size changes with accuracy

Converting the tables above into graphs, we analyze how the loss and accuracy change as the number of epochs increases, depending on the patch sizes. For the loss function, there is an overall decreasing downward trend with all patch sizes. However, the larger patches, such as the 16x16 pixel and 32x32 pixel patches, decrease at a much faster rate. Initially, these larger patches start at a much higher loss, but once reaching epoch 4, it becomes lower than the other patch sizes. However, with accuracy, there's an overall increasing trend as the epoch count increases. The larger patches can increase at a much faster rate compared to the smaller patches, especially shown with the 32x32 pixel patches (green line). Initially, the larger patch sizes start at a lower accuracy than the small ones. When the training cycle reaches 2-3 epochs, because of the rapidly increasing rate, the larger patches result in a larger value than the small patches. This shows how the system can analyze large amounts of information very precisely with minimal errors.

3.2. Accuracy Comparison

We compared the accuracy results from our model to this previously existing research on Optimizing Deeper Spiking Neural Networks.[22] Here is the data table showing a comparison of our experiment's collected data with previous existing research data:

Table 2. Data table of the results from another experiment that implemented SNN models on the DVS128 Gesture Dataset

Method	Accuracy (%)
Spikformer-4-384 400E	95.51
Spikformer-4-256	93.94
STBP NeuNorm	90.53
Diet-SNN	92.54
Our Method (ViT)	98.4

DOI: 10.54254/2755-2721/107/20241399

Table 2 analyzes the accuracies of different SNN models on DVS data using their model Spikformer, which will be our baseline for accuracy comparison. Spikformer classifies DVS data by converting the input images into spike- form feature vectors, in which the Spikformer encoder captures relevant information. This paper analyzes past SNN models to their Spikformer SNN models for accuracy comparison. All of these models perform relatively similar to each other, at 90% or above. The Spikformer-4-384 400E and Spikformer-4-256 have a final accuracy at 95.51% and 93.94% respectively. The other SNN models, like STBP NeuNorm, perform at 90.53%, and Diet-SNN performs at 92.54%. All of the models listed above average at an accuracy of 93.13%.

The accuracy values found through our custom Vision Transformer to our DVS128 Gesture Dataset outperforms the values of the accuracies from previous research that implemented SNNs on DVS data. Looking at the average of all the above SNN models, which is 93.13%, our model performs around 4.37% more, resulting at a final accuracy of 98.4%.

This accuracy can be attributed to the vision transformer's ability to analyze information better when the depth of data increases. After the input is split into individual patches, they are embedded into the transformer in the order that it finds the most efficient.[23] Self-attention mechanisms within the transformer allow it to dedicate its analysis toward a specific portion of the input. The transformer continues this down a single sequence, which means that the transformer analyzes the input one layer at a time; the transformer focuses all of its attention on a portion of the image and then pieces these results together, resulting in a higher overall classification accuracy. As well as analyzing the image patch by patch, the transformer will view the patch in relation to the rest of the image by detecting local and global dependencies, using this information to figure out the context. Only frame-to-frame differences are analyzed, which lowers computational energy. The architecture achieved a high accuracy with high efficiency due to self-attention, which means that biases specific to the input are not required. The transformer performed the best with the largest patch size, 32 x 32, because there were more pixels present that could be used to ascertain the context.

4. Conclusion

In this paper, we proposed implementing a Vision Transformer machine learning architecture to process Dynamic Vision Sensing (DVS) data for increased performance when analyzing and classifying motion sensor data in real- time. Using the vision transformer to analyze the DVS128 gesture dataset, the accuracy was reduced by 50%, having weaker capabilities to classify the different human poses. However, the model shows a positive trend with the accuracy increasing as the number of epochs increases, suggesting that the model has a large potential for highly accurate results. Furthermore, the model reduced computational costs enough to potentially justify widespread use. Across many trials, the extensive training of the program yielded such results. Using this program, there are numerous possibilities to expand our research to various fields of real-time motion-sensing vision systems, including emerging technologies like autonomous vehicles, surveillance systems, or vision classification in robotics.

Acknowledgments

This is a collaborative project performed for a college course in a research program. Thank you to Instructor Rhys Gretsch and the University of Santa Barbara for facilitating this project.

I acknowledge the use of ChatGPT to debug aspects of the transformer model. The prompts include "Tell me what is wrong with this program." This is followed by a portion of the code pasted in. The output for these prompts was implemented back into the program to help solve confusing errors. Generative AI (ChatGPT) was never used in this report; the authors wrote it themselves.

Author contribution statement

A.Y., O.S., and S.K. planned the experiments, A.Y. wrote, conducted, and tested the codes, and A.Y., O.S., and S.K. analyzed the results. All authors reviewed the manuscript.

DOI: 10.54254/2755-2721/107/20241399

References

[1] "What Are the Disadvantages of a Vision Sensor? | IndMALL, " Jun. 25, 2024. https://www.indmall.in/faq/what-are-the-disadvantages-of-a-vision-sensor/#:~:text=In%20factory%20settings%2C%20the%20movement (accessed Jul. 07, 2024).

[2] Photonics Media editors, "Dynamic Vision Sensors Detect, Differentiate Movement in Real Time, " Photonics.com, Aug. 13, 2019. https://www.photonics.com/Articles/Dynamic_Vision_Sensors_Detect_Differentiate/a65017 (accessed Jul. 07, 2024).

[3] "Dynamic Vision Sensors – A Brief Overview - Image Sensor TechStream Blog | TechInsights, " www.techinsights.com.https://www.techinsights.com/blog/image-sensor/dynamic-vision-sensors-brief-overview-image-sensor-techstream-blog#:~:text=Dynamic%20Vison%20Sensors%20are%20Asynchronous (accessed Jul. 07, 2024).

[4] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. arXiv preprint arXiv:2010.11929, 2020.

[5] A. M. George, D. Banerjee, S. Dey, A. Mukherjee and P. Balamurali, "A Reservoir-based Convolutional Spiking Neural Network for Gesture Recognition from DVS Input, " 2020 International Joint Conference on Neural Networks (IJCNN), Glasgow, UK, 2020, pp. 1-9, doi: 10.1109/IJCNN48605.2020.9206681.

[6] A. M. George, D. Banerjee, S. Dey, A. Mukherjee and P. Balamurali, "A Reservoir-based Convolutional Spiking Neural Network for Gesture Recognition from DVS Input, " 2020 International Joint Conference on Neural Networks (IJCNN), Glasgow, UK, 2020, pp. 1-9, doi: 10.1109/IJCNN48605.2020.9206681.

[7] W. He et al., Eds., "Comparing SNNs and RNNs on neuromorphic vision datasets: Similarities and differences, " Neural Networks, Dec. 2020. https://www.sciencedirect.com/science/article/pii/S0893608020302902#sec5 (accessed Jul. 07, 2024).

[8] R. Graça, B. Mcreynolds, and T. Delbruck, "Shining light on the DVS pixel: A tutorial and discussion about biasing and optimization." Accessed: Jul. 08, 2024. [Online]. Available: https://openaccess.thecvf.com/content/CVPR2023W/EventVision/papers/Graca_Shining_Light_on_the_DVS_Pix el_A_Tutorial_and_Discussion_CVPRW_2023_paper.pdf

[9] L. Meng et al., "AdaViT: Adaptive Vision Transformers for Efficient Image Recognition." Accessed: Jul. 16, 2024. [Online]. Available: https://openaccess.thecvf.com/content/CVPR2022/papers/Meng_AdaViT_Adaptive_Vision_Transformers_for_Effi cient_Image_Recognition_CVPR_2022_paper.pdf

[10] Y. Kim and P. Panda, "Optimizing Deeper Spiking Neural Networks for Dynamic Vision Sensing, " Neural Networks, vol. 144, pp. 686–698, Dec. 2021, doi: https://doi.org/10.1016/j.neunet.2021.09.022.

[11] O. Moutik et al., "Convolutional Neural Networks or Vision Transformers: Who Will Win the Race for Action Recognitions in Visual Data?, " Sensors, vol. 23, no. 2, p. 734, Jan. 2023, doi: https://doi.org/10.3390/s23020734.

[12] M. Raghu, T. Unterthiner, S. Kornblith, C. Zhang, and A. Dosovitskiy, "Do Vision Transformers See Like Convolutional Neural Networks?" Available: https://proceedings.neurips.cc/paper_files/paper/2021/file/652cf38361a209088302ba2b8b7f51e0-Paper.pdf

[13] "Getting started — 1.4.3, " tonic.readthedocs.io. https://tonic.readthedocs.io/en/latest.

[14] PyTorch, "PyTorch documentation — PyTorch master documentation, " Pytorch.org, 2019. https://pytorch.org/docs/stable/index.html

[15] Y. Kim and P. Panda, "Optimizing Deeper Spiking Neural Networks for Dynamic Vision Sensing, " ScienceDirect, Dec. 2021. https://www.sciencedirect.com/science/article/pii/S0893608021003841

[16] "Papers with Code - DVS128 Gesture Dataset, " paperswithcode.com. https://paperswithcode.com/dataset/dvs128-gesture-dataset

DOI: 10.54254/2755-2721/107/20241399

[17] "Event-based Vision Sensor(EVS)Technology | Technology, ” Sony Semiconductor Solutions Group. https://www.sony-semicon.com/en/technology/industry/evs.html
[18] "Event-based Vision Sensor(EVS)Technology | Technology, ” Sony Semiconductor Solutions Group. https://www.sony-semicon.com/en/technology/industry/evs.html
[19] https://github.com/anniey-17/Integrating_DVS_with_ViT_for_enhanced_real_time_processing_and_classification/tree/main
[20] “Accuracy Test definition | Psychology Glossary | AlleyDog.com, ” www.alleydog.com. https://www.alleydog.com/glossary/definition.php?term=Accuracy+Test#:~:text=An%20accuracy%20test%20is%2 0designed (accessed Jul. 08, 2024).
[21] neuralthreads, “Categorical cross-entropy loss — The most important loss function, ” Medium, Dec. 26, 2021. https://neuralthreads.medium.com/categorical-cross-entropy-loss-the-most-important-loss-function- d3792151d05b
[22] Z. Zhou et al., “SPIKFORMER: WHEN SPIKING NEURAL NETWORK MEETS TRANSFORMER.” Accessed: Jul. 25, 2024. [Online]. Available: https://arxiv.org/pdf/2209.15425v2
[23] G. Boesch, “Vision Transformers (ViT) in Image Recognition - 2024 Guide, ” viso.ai, Nov. 25, 2023.https://viso.ai/deep-learning/vision-transformer-vit/#:~:text=The%20visual%20transformer%20divides%20an