VisionNet: Investigating the Influence of Brain-Inspired Features on CNNs for the task of Image Classification

Quoc Anh (Alan) Bui **Anshumaan Chauhan**
University of Massachusetts Amherst
College of Information & Computer Sciences
{qhbui,achauhan}@umass.edu

Abstract

There are several brain-inspired characteristics that have gained a lot of popularity in the past few years, either due to their ability of performing computations efficiently - spiking neurons, or because of their better performance on tasks - attention mechanism. In the field of computer vision, the way Convolutional Neural Networks (CNNs) processes the images differs significantly from how a brain process a vision. We perform an investigative study, that incorporates brain-inspired features such as i) Attention, ii) Multi-Feature Extraction and iii) Lateral Connections into a CNN architecture and observe the affects of these features on the performance metrics (accuracy) of the architecture on the task of image classification. Experiments show that, brain-inspired characteristics in the architecture lead to improvement in performance on the task of image classification on CIFAR10 and CIFAR100 by 1.6% and 3.35% respectively.

1 Introduction

Recently, there has been an increase in the use of deep learning models in several tasks because of their exceptionally well performance on tasks such as Image Recognition, Image Classification and many more. More specifically, Convolutional Neural Networks (CNNs) is the most used algorithm for several computer vision tasks. As compared to the previous models such as Multi-layered perceptrons (MLPs), CNNs have shown better performance, with much less number of parameters and comparatively easier training methodology.

However, the neurons that are used in the CNN architecture are very simple and significantly different from the biological neurons in broadly three following aspects Samadi et al. (2017):

- The outputs of brain inspired neurons are spikes rather than a computed real value (calculated through mathematical operations).

34th Conference on Neural Information Processing Systems (NeurIPS 2020), Vancouver, Canada.

- Brain neurons are not linked by some mathematical formula (multiplication followed by an activation function) unlike CNN neurons. Instead the neurons in brain are related dynamically to each other.
- Lastly, there is no solid evidence/proof which confirms the fact that neurons in brain also learn new things using the concept of backpropagation.

Current research focuses on the impact of brain-inspired characteristics like spiking neurons, continual learning, replay, and attention on neural networks. The self-attention mechanism, initially used in Natural Language Processing with Transformers, has been adapted for Computer Vision tasks, offering advantages such as the ability of capture long-range dependencies and training parallelization, observed in NLP.

Various CNN architectures have tried to include some of the features (but not more than one). Thus, our project aims to enhance a simple CNN architecture by incorporating three brain-inspired characteristics (Attention, Multi-feature extraction, and Lateral connections) and evaluating their impact on Image Classification. The report is organized as follows: Section 1 provides an overview of the project, Section 2 reviews relevant literature, Sections 3 and 4 cover the Theoretical background and Proposed methodology respectively, Section 5 presents the experimentation results, and Section 6 discusses about the implementation challenges and the conclusion. Finally, Section 7 points out some of the possible future works on this project.

2 Literature Survey

2.1 Spiking Neural Networks (SNNs)

Spiking Neural Networks are the neural network models that are composed of spiking neurons. These spiking neurons are highly energy efficient Wang et al. (2023) and this advantage is mainly due to their multiplication free property that they exhibit because of the use of binarized intermediate activation.

First model of SNNs was proposed in 1997, and since then a lot of focus has been given to integration of SNNs in the Deep Convolutional architectures, to improve the efficiency and the performance of the architectures on the different tasks in Computer Vision Li et al. (2022).

New advance research are trying to implement different types of spiking methodologies. For example, a research Hazan et al. (2018) was performed on the coarse scale approximations of spiking neurons to evolve their system in order of the refractory period. Another research explored the mathematical mapping that existed between the biological parameters of the LIF neurons and the Rectified Linear Unit (ReLU) activations Lu and Xu (2022).

A efficient MLP design which uses spiking neurons Leng et al. (2023) along with skip (or residual) connections and spiking patch encoding layer showed competetive performance on ImageNet dataset in comparison to models like Spiking ResNet and Spiking VGG. It made use of MFI friendly bacth normalization along with a MLP-Mixer Tolstikhin et al. (2021) architecture to increase the performance of the model.

However, SNNs face the following drawbacks:

- Quality degradation caused by the loss of information
- Requires a lot of preprocessing of the data to make it compatible to use with SNNs

- Hardware restrictions - most used Machine Learning frameworks such as PyTorch Paszke et al. (2019), TensorFlow Abadi et al. (2016), Caffe Jia et al. (2014) and JAX Frostig et al. (2018) do not offer efficient convolutional operations with 0/1 activations. Not a big limitation as there are a lot of libraries proposed that are built in Python on top of previously mentioned frameworks for the simulation of spiking neurons for the task of machine learning and reinforcement learning such as BindsNet Benali Amjoud and Amrouch (2020), Brain2GeNN Stimberg et al. (2018) and PyNN Davison et al. (2009).

2.2 Brain-Inspired Neural Networks

Image aesthetic assessment involves rating an image based on specific rules and demonstrated features. A brain-inspired neural network which learns image attributes from different feature maps was proposed Wang et al. (2016). Once the model is pretrained in a parallel pathway setting, it associates these pathways with a synthesis network and fine-tunes it using human ratings like a reward model, thus outperforming CNNs.

The challenges in image classification include vulnerability to adversarial inputs and the issue of catastrophic forgetting in machine learning models. To address these challenges, a neuro-inspired CNN architecture Huang et al. (2019) integrates top-down and bottom-up pathways using a feedback generative network, providing robustness against adversarial inputs such as noise, occlusion, and blur. Additionally, a replay-based framework Van de Ven et al. (2020) tackles catastrophic forgetting by incorporating Brain-Inspired Replay and Synaptic Intelligence in a class-incremental learning setting, achieving performance close to the Joint training approach.

Sparse Neural Networks offer efficiency without sacrificing accuracy, but suffer from challenges like poor sparsity performance and dense gradient computation. Hebbian Learning Atashgahi et al. (2022), a brain-inspired approach, improves training of sparse networks, outperforming SOTA architectures on datasets like MNIST, CIFAR10, and CIFAR100.

2.3 Brain-Inspired Capabilities in the State-of-the-art (SOTA) Networks

There exists a lot of CNN architectures that are not explicitly mention about any brain inspired characteristics, but they are inherently present in these architectures, and the results have shown that they have benefit in one or the other way from these characteristics. Some of such famous SOTA architectures are:

1. **DenseNet** Huang et al. (2017) utilizes lateral connections between layers, resulting in improved performance, resolved vanishing gradient issues, enhanced feature contextualization, and parameter reduction.
2. **ResNet**He et al. (2016) addresses depth and vanishing gradient issues by incorporating lateral connections in its architecture, enabling information integration from previous layers.
3. **Xception**Chollet (2017) extends the Inception model by employing depthwise separable convolutional layers followed by a point-wise convolutional operation instead of a convolutional layer. It captures the brain-inspired characteristic of multi-feature extraction.
4. **Visual Transfomers:** Attention mechanism helps us to focus on the most important aspect of the information based on the environment and our current behavior. In theory, there are different types of attention mechanisms available - soft attention, hard attention and self-attention de Santana Correia and Colombini (2022). Visual transformers combine this self-attention concept with Convolutional Neural Networks - and have shown competetive

performance with SOTA CNN architectures (but at the cost of more parameters and more training time).

3 Convolution enhanced with Self-Attention (Theoretical Background)

3.1 Convolution Revisit

The convolution operation (2D) involves sliding a small filter or kernel $K \in \mathbb{R}^{C \times FH \times FW}$ with learnable parameters over the input tensor $X \in \mathbb{R}^{C \times H \times W}$, computing a dot product between the filter and the portion of the input tensor it is currently overlaying, and producing a single output value:

$$Y_{ij} = \sum \left(K \odot \hat{X} \right) + b \tag{1}$$

where $Y \in \mathbb{R}^{H' \times W'}$, $\hat{X}$ is the current portion of the input tensor and $\odot$ is element-wise multiplication.

The output of the convolution operation is a 2D tensor called a feature map. The feature map represents the activation of a set of filters at a given spatial location in the input image. The output of a convolution layer resulted in a tensor $Y \in \mathbb{R}^{C' \times H' \times W}$ where the number of filters determines the depth or number of output channels C'. Convolution layers enable the network to learn local features, such as edges and textures, by detecting patterns in a local neighborhood of the input image via sliding filters.

3.2 Self-Attention Revisit

Self-attention mechanism is a key component of transformers that allows them to process input sequences of variable length and capture the long-term dependencies between the different elements in the sequence. In next-word prediction, the prefix words are tokenized, and the self-attention mechanism is applied at each token to compute the significance of previous tokens for predicting the next one.

Similarly, in our image classification problem, the self-attention mechanism can be utilized to evaluate the importance of a pixel in the image relative to all other pixels.

3.2.1 Query, Key and Value

The self-attention mechanism Vaswani et al. (2017) consists of queries and keys of dimension d_k, and values of dimension d_v. It then computes the dot products of the query with all keys, divide each by $\sqrt{d_k}$, and apply a softmax function to obtain the weights on the values. In practice, it computes the attention function on a set of queries simultaneously, packed together into a matrix W_q. The keys and values are also packed together into matrices W_k and W_v . We compute the matrix of outputs as following:

$$\text{Attention}(W_q, W_k, W_v) = \text{softmax}\left(\frac{W_q \cdot W_k^T}{\sqrt{d_k}} \right) \cdot W_v \tag{2}$$

The attention weight determines how much of a value from a particular token can contribute to the final prediction, enabling the model to selectively attend to relevant parts of the input sequence and capture complex relationships.

4 Methodology

4.1 Integration of self-attention into convolution

In this research, we experiment with 3 architectures where they differ the most in the attention weights computation.

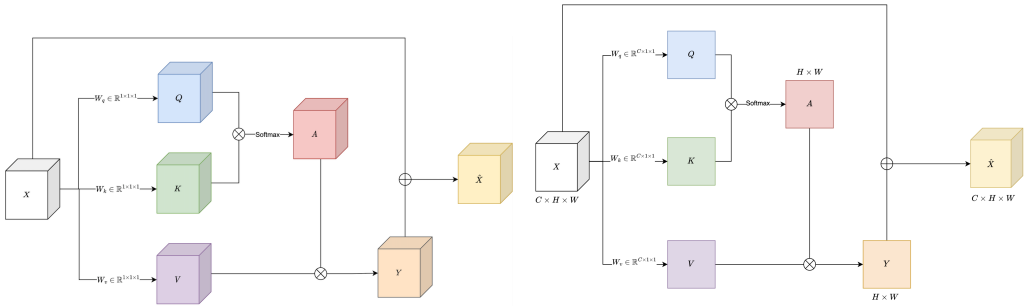


Figure 1: Attention-CNN Architecture 1 (left) and Architecture 2 (right)

Architecture 1 and 2 shares lot of similarities where architecture 1's attention weights determine the importance of pixel i w.r.t all the pixels across the channels of the image and architecture 2 simply does similar job, but excludes the importance of channels, i.e, all pixels (i, j, c) where $c : 1 \dots C$ are assigned the same weight. One interesting point of architecture 1 is that it uses $1 \times 1 \times 1$ kernel, allowing to reuse the same kernels over all convolution layers, mimicking the action of sharing parameters in the original self-attention mechanism where W_q, W_k and W_c is shared when computing q, k and v among all the tokens.

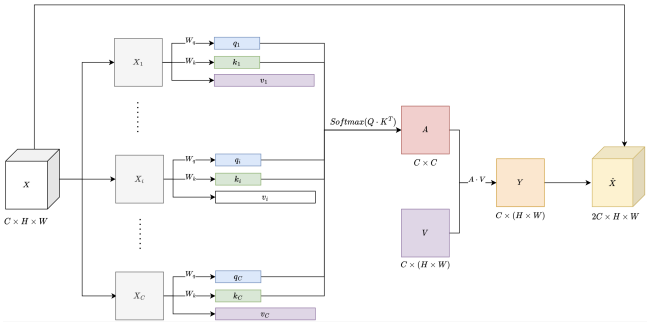


Figure 2: Attention-CNN Architecture 3

Architecture 3 on the other hand points the direction to the feature maps instead of pixels, i.e assigning relative weight to each feature map w.r.t to all the feature maps at each convolution layer. We can think of each feature map as a token $\in \mathbb{R}^{H \times W}$. After that, $W_q, W_k \in \mathbb{R}^{H \times W, D}$ are used to computed q and k. It is worth noticing that v is kept the same as the token.

4.2 Residual Connection

The Transformer architecture utilizes residual connections in both the multi-head attention and feedforward layers. Residual connections address the vanishing gradient problem by adding the input to the output, enabling smoother gradient flow during backpropagation. This enhances model stability, convergence, and performance in natural language processing tasks.

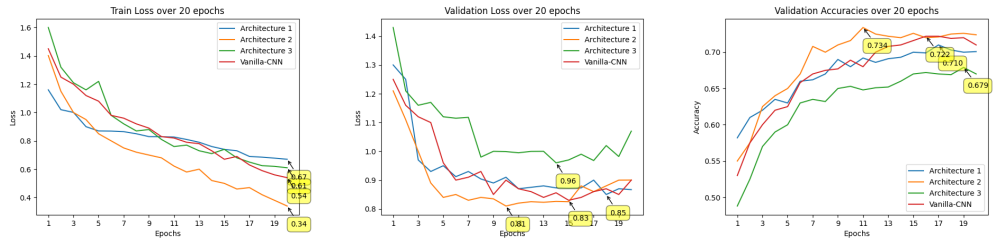
The output of self-attention mechanism Y is element-wise added in architecture 1 and 2 and appended in architecture 3 to X (the output of the convolution layer) in order to produce the final output $\hat{X}$ for the attention-convolution layer.

5 Experimentation and Results

We trained both vanilla CNN and Attention-CNN(s) on CIFAR10 and CIFAR100 datasets for over 20 epochs with various optimizers and learning rates. We discovered that Adam optimizer with learning rate `lr = 1e-3` significantly outperforms other settings. Thus, we decided to omit other results and focus on only this setting.

Both the Attention-CNN and vanilla CNN architectures consist of four convolution layers followed by batch normalization, ReLU activation function, and an optional pooling layer. Furthermore, both architectures employ a classifier comprising three hidden layers. A distinctive feature of Attention-CNNs is the incorporation of a self-attention mechanism immediately after the output of each convolution layer, as depicted in Figures 1, 2, and 3.

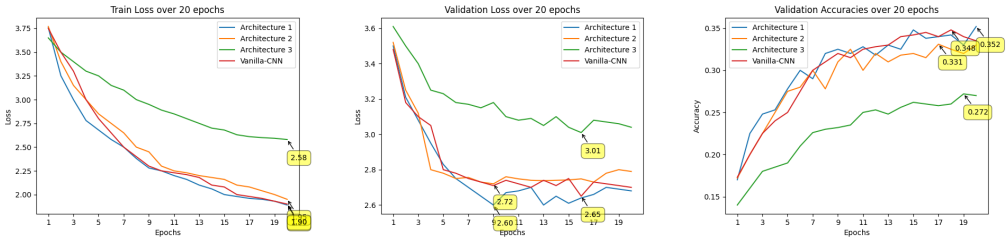
5.1 CIFAR10



(a) Training Loss Curves (b) Validation Loss Curves (c) Validation Accuracy Curves

Figure 3: Performance of Proposed architectures and Vanilla CNN on CIFAR10

5.2 CIFAR100



(a) Training Loss Curves (b) Validation Loss Curves (c) Validation Accuracy Curves

Figure 4: Performance of Proposed architectures and Vanilla CNN on CIFAR100

5.3 Test Accuracy

Our project aims to investigate the influence of brain-inspired features on CNNs for the task of image classification. Therefore, besides training on benchmark between the four models we also train several other baseline models such as VGG16, ResNet50 and Xception. These models are trained using two different settings - vanilla fine-tunning and transfer learning. The test accuracies of all the models on both CIFAR10 and CIFAR100 are recorded in Table 1.

Model/Dataset	CIFAR10				CIFAR100			
	Vanilla FT		Transfer Learning		Vanilla FT		Transfer Learning	
Epochs	5	10	5	10	5	10	5	10
VGG16	10.00	12.43	64.33	65.67	76.51	76.85	64.34	66.18
ResNet 50	55.60	70.42	27.86	26.33	66.31	76.10	29.68	30.75
Xception	88.63	89.83	70.03	69.83	89.49	88.86	69.72	70.69
Traditional CNN	71.28				33.15			
Attention-CNN	71.6/72.8/65.85				35.2/36.5/27.29			

Table 1: Test Accuracy Summary

6 Discussion

Upon analyzing the results, it becomes evident that both architecture 1 and architecture 2 of the Attention-CNN and vanilla CNN exhibit relatively similar performance. Additionally, Architecture 2 demonstrates both the advantages of brain-inspired features - faster training and better accuracy. The subtle disparities observed can be attributed to random weight initialization and sub-optimal implementation of self-attention mechanism. However, a significant distinction arises when comparing the performance of architecture 3 to the others. In all metrics, architecture 3 demonstrates comparatively poorer performance. This discrepancy prompted us to invest further and conduct a thorough analysis, specifically comparing its performance to that of the vanilla CNN architecture through the examination of heat maps via Grad-CAM.

6.1 Heat Maps

In CNNs, a heat map is a visual representation that highlights the important regions of an image. It shows the areas with higher activation values, indicating their significance in the network's decision-making. Heat maps help analyze CNN behavior and understand where the model focuses its attention within an image. They provide insights into network performance, important features, and decision-making processes.

We examined in total of five samples to compare the prediction performance between Attention-CNN (architecture 3) and vanilla CNN. In the first three samples, Attention-CNN provided accurate predictions while vanilla CNN did not and vice versa for the other 2 samples. To gain further insights, we conducted a heatmap analysis on the kernel of the fourth convolutional layer, the results of which are illustrated below:

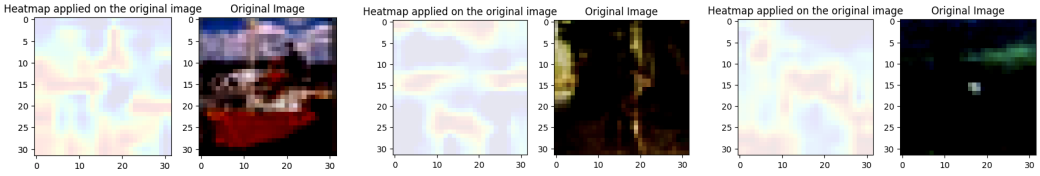


Figure 5: Three samples where Attention-CNN outperforms vanilla CNN

In cases where Attention-CNN outperforms vanilla CNN, the depicted objects were not evidently distinguishable. Through the analysis of the heatmaps, it becomes apparent that numerous regions within the images significantly influenced the model's decision-making process. This suggests that the model effectively considered combinations of feature maps that highlighted various areas of the images in order to reach a final decision. Conversely, Attention-CNN exhibited clear shortcomings in

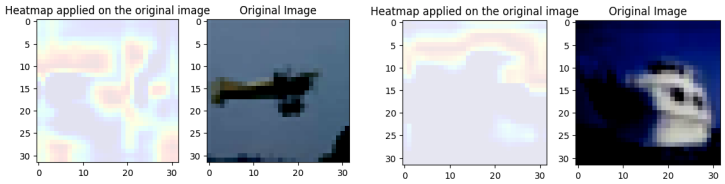


Figure 6: Two samples where Attention-CNN under performs vanilla CNN

capturing the crucial features of the objects in the images, such as the shape of the plane and the boat. This behavior makes it unclear about the performance drop being due to a shift from pixel level self-attention to a feature map based self-attention.

6.2 Conclusion

The evaluation of Attention-CNN architectures 1 and 2 yields inconclusive results. The minimal performance variation can be attributed to random weight initialization as well as non optimal self-attention implementation. In architecture 1, the parameters for W_q, W_k, and W_v are singular values. In architecture 2, the dimensional mismatch between the outputs of different convolution layers prevents sharing of W_q, W_k, and W_v. Additionally, the process of element-wise addition between the output of the self-attention mechanism and the convolution layer's output has the potential to introduce unintended "noises" as opposed to achieving the intended purpose of residual connection for subsequent convolution layers.

In the case of Attention-CNN architecture 3, the inclusion of the self-attention mechanism has a disruptive effect on the overall function of the convolution layers, resulting in a decline in the model's performance. The model fails to effectively capture the crucial features that should have been highlighted. However, it is worth noting that there are certain instances where Attention-CNN exhibits performance that aligns with its intended behavior.

7 Future Work

This experimental research serves as an initial exploration to assess the suitability of incorporating self-attention in computer vision tasks. Further investigations are required to develop an efficient attention-convolution fused model. This necessitates conducting extensive experiments and research to design the model, particularly focusing on the attention weight and (q, k, v) computations, the ability to share W_q, W_k and W_v weight matrices among all the convolution layers and the inclusion of multi-head self-attention Vaswani et al. (2017).

On the other hand, we do not rule out the possibility that self-attention cannot be fused with convolution. This comes from the nature of convolution and self-attention is much different from each other and used they are used with different setups. Vision Transformers (ViTs) for instance divide the input image into fixed-size non-overlapping patches, which are then linearly embedded to obtain patch embeddings, along with positional embeddings, as input to the transformer encoder. They then capture global relationships and dependencies in images by leveraging self-attention mechanisms. ViTs have achieved remarkable performance in image classification and other vision tasks, often rivaling or surpassing CNN-based models.

References

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., et al. (2016). Tensorflow: a system for large-scale machine learning. In *Osdi*, volume 16, pages 265–283. Savannah, GA, USA.

Atashgahi, Z., Pieterse, J., Liu, S., Mocanu, D. C., Veldhuis, R., and Pechenizkiy, M. (2022). A brain-inspired algorithm for training highly sparse neural networks. *Machine Learning*, pages 1–42.

Benali Amjoud, A. and Amrouch, M. (2020). Convolutional neural networks backbones for object detection. In *Image and Signal Processing: 9th International Conference, ICISP 2020, Marrakesh, Morocco, June 4–6, 2020, Proceedings 9*, pages 282–289. Springer.

Chollet, F. (2017). Xception: Deep learning with depthwise separable convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1251–1258.

Davison, A. P., Brüderle, D., Eppler, J. M., Kremkow, J., Muller, E., Pecevski, D., Perrinet, L., and Yger, P. (2009). Pynn: a common interface for neuronal network simulators. *Frontiers in neuroinformatics*, page 11.

de Santana Correia, A. and Colombini, E. L. (2022). Attention, please! a survey of neural attention models in deep learning. *Artificial Intelligence Review*, 55(8):6037–6124.

Frostig, R., Johnson, M. J., and Leary, C. (2018). Compiling machine learning programs via high-level tracing. *Systems for Machine Learning*, 4(9).

Hazan, H., Saunders, D. J., Khan, H., Patel, D., Sanghavi, D. T., Siegelmann, H. T., and Kozma, R. (2018). Bindsnet: A machine learning-oriented spiking neural networks library in python. *Frontiers in neuroinformatics*, 12:89.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Huang, G., Liu, Z., Van Der Maaten, L., and Weinberger, K. Q. (2017). Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708.

Huang, Y., Dai, S., Nguyen, T., Bao, P., Tsao, D. Y., Baraniuk, R. G., and Anandkumar, A. (2019). Brain-inspired robust vision using convolutional neural networks with feedback. In *Real Neurons {\&} Hidden Units: Future directions at the intersection of neuroscience and artificial intelligence@ NeurIPS 2019*.

Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., Guadarrama, S., and Darrell, T. (2014). Caffe: Convolutional architecture for fast feature embedding. In *Proceedings of the 22nd ACM international conference on Multimedia*, pages 675–678.

Leng, L., Li, B., Cheng, R., Shen, S., Zhang, K., Zhang, J., and Liao, J. (2023). Rethinking deep spiking neural networks: A multi-layer perceptron approach.

Li, W., Chen, H., Guo, J., Zhang, Z., and Wang, Y. (2022). Brain-inspired multilayer perceptron with spiking neurons. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 783–793.

Lu, S. and Xu, F. (2022). Linear leaky-integrate-and-fire neuron model based spiking neural networks and its mapping relationship to deep neural networks. *Frontiers in neuroscience*, page 1368.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al. (2019). Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32.

Samadi, A., Lillicrap, T. P., and Tweed, D. B. (2017). Deep learning with dynamic spiking neurons and fixed feedback weights. *Neural computation*, 29(3):578–602.

Stimberg, M., Goodman, D. F., and Nowotny, T. (2018). Brian2genn: a system for accelerating a large variety of spiking neural networks with graphics hardware. *bioRxiv*, page 448050.

Tolstikhin, I., Houlsby, N., Kolesnikov, A., Beyer, L., Zhai, X., Unterthiner, T., Yung, J., Steiner, A., Keysers, D., Uszkoreit, J., Lucic, M., and Dosovitskiy, A. (2021). Mlp-mixer: An all-mlp architecture for vision.

Van de Ven, G. M., Siegelmann, H. T., and Tolias, A. S. (2020). Brain-inspired replay for continual learning with artificial neural networks. *Nature communications*, 11(1):4069.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomes, A. N., Kaiser, L., and Polosukhin, I. (2017). Attention is all you need.

Wang, X., Zhang, Y., and Zhang, Y. (2023). Mt-snn: Enhance spiking neural network with multiple thresholds. *arXiv preprint arXiv:2303.11127*.

Wang, Z., Chang, S., Dolcos, F., Beck, D., Liu, D., and Huang, T. S. (2016). Brain-inspired deep networks for image aesthetics assessment. *arXiv preprint arXiv:1601.04155*.