Friendly introduction to Spikformers: how self attention modules can be integrated into Spiking Neural Networks

Isabel Prieto Payo, Víctor Gutiérrez García, *Escuela Técnica Superior de Ingerios de Telecomunicación, Universidad Politécnica de Madrid*

***Abstract*—Neurological modelling has been used since many decades to make artificial systems improve their performance in diverse cognitive tasks. Spiking Neural Networks (SNNs) are a visible example of how a realistic model of information transference in neurons can be introduced in Artificial Neural Network (ANN) architectures. This modelling requires input data to resemble a train of spikes, making recurrent architectures the most extended choice to deal with it. This work aims to reveal how the advances in sequential data processing introduced by Transformer architectures and Self-Attention methods can be applied to improve and enrich SNN implementations, giving examples of the achieved performance.**

***Index Terms*—Neurological inspiration, sequential data, Spiking Neural Network (SNN), Self-Attention, Transformers, Spikformers**

I. Introduction

ARTIFICIAL General Intelligence (AGI) refers to those cognitive capabilities that enable the system to whom they belong to outperform in a wide range of tasks. A rigorous definition is still not be given by the research community, but some criteria such as the Turing test [1] have been proposed to evaluate the cognitive abilities of a system.

In the effort to simulate human behavior, significant emphasis has been placed on transferring neurological principles to artificial models. Consequently, the way neurons communicate with each other has gained considerable attention. In a neuron, dendrites are structures specialized in receiving information, with transmission driven by a sequence of electrical impulses or action potentials. These impulses are generated by the depolarization of the neuron's membrane through its ion channels. The resulting impulses, or spikes, are transmitted along the neuron's axon to the presynaptic terminals, where the connection between neurons (synapses) occurs.

Spiking Neural Networks (SNNs) are a remarkable result of the attempts on modelling neurons behavior in a realistic way, and are based on the Leaky Integrate-Fire (LIF) principle, that mimics the action potential process in neurons. This involves handling sequential data and therefore the associated optimization process can be treated similarly as the one of Recurrent Neural Networks (RNNs). However, even if the recurrent approach is the most consolidated one for the deployment of SNNs-based models, recent breakthroughs in the context of sequence-based problems and memory systems that rely on historical patterns are eligible to be applied to build SNN architectures. Specifically, this work explores how self-attention mechanisms can be used within SNN architectures to improve its performance.

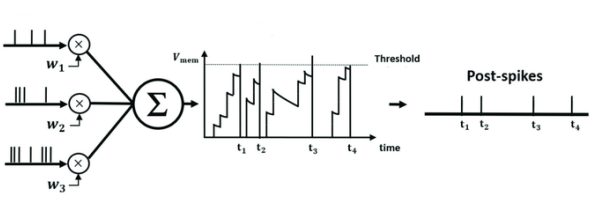

Fig. 1. Leaky Integrate-Fire (LIF) process representation. At the right, spikes coming from previous neurons ($X[t+1]$) are integrated in the LIF Neuron. At the middle, the concatenation of decaying potential, spikes-integration and potential resetting is depicted through time. At the left, spikes that are output by the LIF Neuron are represented aligned with the firing/potential resetting events.

The rest of the paper is organized as follows. Section 2 is dedicated to reviewing the core concepts of this work, namely SNNs and Transformers; Section 3 presents how these two frameworks interact together to give rise to Spikformers; and Section 4 is dedicated to a discussion and drawing the main conclusions.

II. Review on Transformers and Spiking Neural Networks

A. Spiking Neural Networks

The Leaky Integrate-and-Fire (LIF) (figure 1) model for SNNs Neurons is the more extended one due its trade-off between accuracy to real neurons and applicability. It consists of three steps in which the potential $U(t)$ evolves through time: in the Leaky step the potential evolves according to a differential equation; in the Integration step the potential increases by S when the Neuron receives a spike; and in the Fire step the neuron potential is reset only when it goes beyond a threshold U_{th}. This process can be summarized in the following equation, unveiling the recurrent nature of the architecture.

$$U[t+1] = \beta U[t] + WX[t+1] - S[t]U_{th} \quad (1)$$

This work was supported by the Universidad Politécnica de Madrid. Finally, the authors would like to thank Escuela Técnica Superior de Ingenieros (ETSI) de Telecomunicación for its indirect support.

Isabel Prieto Payo and Victor Guitierrez García are with Escuela Técnica Superior de Ingenieros (ETSI) de Telecomunicación, Universidad Politécnica de Madrid, 28040, Madrid, Spain (e-mail: isabel.prieto.payo@alumnos.upm.es, v.ggarcia@alumnos.upm.es)

Manuscript submited in May 20, 2024.

where $S[t]$ equals to one if in a previous neuron it holds $U[t] > U_{th}$ and therefore it emits an spike, W is the learnable parameter and β is an hyperparameter that defines the decay rate.

However, due to the recurrent nature of the problem, some of the limitations of RNN architectures may be transferred to the LIF-based SNN models. Some relevant examples of these limitations are: difficulties in capturing long-term dependencies due to vanishing gradients problem and lack of parallelization capabilities due to sequence processing.

B. Transformers and self-attention mechanisms

The Transformer architecture was first described in [2] featuring an encoder-decoder structure with self-attention mechanisms to process sequential data in a parallel way. This is achieved with the so-called "Scaled Dot-Product Attention".

$$Attention(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right) \tag{2}$$

where Q and K are matrices that pack together a set of queries and keys, respectively, of dimension d_k, and V stacks values of dimension d_v.

Then, the output of the self-attention layer represents the influence of each element of the processed sequence on the representation of the rest of the elements, and it is computed as follows.

$$output = Attention(Q, K, V)V \tag{3}$$

Note that the matrices Q, K and V are learned during the training process thanks to the inputs and the trainable parameters W^Q, W^K and W^V, respectively.

In addition to the self attention mechanism, multi-head attention is explored in the Transformer architecture, linearly projecting the queries, keys and values h times with different, learned linear projections.

$$head_i = Attention(QW_i^Q, KW_i^K, W_i^V) \tag{4}$$

With the described framework, computational complexity is reduced, parallelization is enabled and long-range dependencies detection capabilities are improved, yielding to an improvement over the existing best results.

III. Integration of Self-Attention into SNN

As we have seen in the previous section, one of the modules introduced by transformers is the so-called "Self-Attention". In this proposal we suggest the use of this self-attention mechanism, adapted to SNN, and we obtain the Spiking Self Attention (SSA) variant of the original module. The architecture of the spikformer can be seen in the 2.

Let's dive step by step in the architecture show in 2 and presented in [3]:

1) **Input Image**: We will input a 2D image, represented as **I**.
2) **Spiking Patch Splitting (SPS)**: Is in charge of transforming the 2D image into a D-dimensional vector of spike-form features and splits it into a sequence of N flattened spike-form patches, named x.

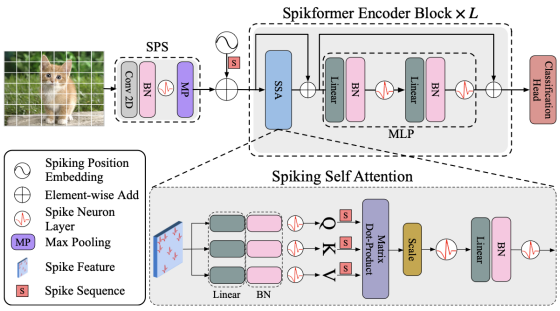

Fig. 2. The overview of Spiking Transformer (Spikformer), which consists of a spiking patch splitting module (SPS), a Spikformer encoder and a Linear classification head. We empircally find that the layer normalization (LN) does not apply to SNNs, so we use batch normalization (BN) instead

$$x = SPS(I) \tag{5}$$

3) **Spiking Position Embedding**: Since float-point position embedding cannot work in SNN we employ a conditional positional embedding which creates spike-form relative position embeddings (RPE). This conditional position embedding incluesd: 2D convolution layer with a kernel of size 3, a Batch Normalizatio and a Spike Neuron Layer (SN). At this point we are at the input of the encoder of the spikeformer, with X_0

$$RPE = SN(BN(Conv2D(x))) \tag{6}$$

$$X_0 = x + RPE \tag{7}$$

4) **Spikformer encoder**: Consists of multiple L-blocks, as in Visual Trasformers (ViT). Each block is composed of:
   - Spiking Self Attention: This is the main component of the Spikformer encoder. It models local and global image information using spike-form Query(Q), Key(K) and Value(V) without using softmax.
   - Multilayer Perceptron block.
   - Residual connections are applied inboth SSA and MLP block.

   At the output of the encoder we have:

$$X_l^{'} = SSA(X_{l-1}) + X_{l-1} \tag{8}$$

$$X_l = MLP(X_l^{'}) + X_l^{'} \tag{9}$$

5) **Output**: After processing, a global average-pooling (GAP) is applied to the feature from the Spikformer encoder. This results in a D-dimensional feature which is sent to the classification head to produce the prediction.

$$Y = CH(GAP(X_L)) \tag{10}$$

Let's dive a little more in SPS and SSA module.

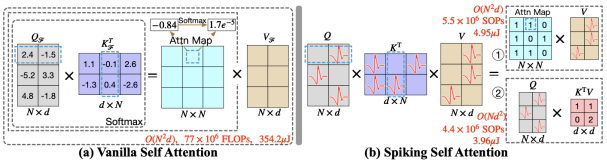

Fig. 3. Illustration of Vanilla Self-Attention (VSA) and Spiking Self-Attention (SSA) mechanisms.

A. Spiking Patch Splitting

As said before, this modules aim to linearly project the input image, I, into a D dimensional spike-form feature and the split this feature into patches with a fixed size. The inductive bias is introduced in the SPS block via convolution layer. For a given image I, we have at the input of the conditional position embedding:

$$x = MP(SN(BN(Conv2d(I)))) \tag{11}$$

Where Conv2D is a 2D convolution layer and MP is a max-pooling layer. The 2D-max-pooling layer is applied to down-sample the feature size after the reception of a SPS block, which has a fixed size. The number of SPS block can be more than 1. When multiple blocks are used, the number of output channels at each block increases gradually until it reach the dimensions of the output embedding. For example, if we had 4 SPS blocks, the channel output of each block would be D/8, D/4, D/2 and D.

After processing the image with the SPS module, the image is split into a patch sequence x.

B. Spiking Self Attention Mechanism

We have seen before how does the Vanilla Self Attention (VSA)mechanism works. We are going to address two main problems of why we cannot use it for SNN.

1) **Incompatible calculations**: the use of float-point matrix multiplication and a softmax function, which involves complex operations like exponentiation and division are not suitable for how SNNs work.
2) **Computational inefficiency**: The quadratic space and time complexity the Self-Attention mechanism requires does not match the requirements of the SNN.

In the proposed Spiking Self Attention Mechanism some efficiency improvements are introduced. A comparison between the VSA and SSA mechanisms is shown in 3

In this implementation, the query(Q), key(K) and value(V) are computed through learnable matrices and then they become spiking sequence using different spike neuron layers, as shown below:

$$Q = SN_Q(BN(XW_Q)) \tag{12}$$

$$K = SN_K(BN(XW_K)) \tag{13}$$

$$V = SN_V(BN(XW_V)) \tag{14}$$

Where W_Q, W_K and W_V are learnable matrices and X is the input. The query and value only contain 0 and 1, giving a spike-form nature to the matrix and making it sparse. This will reduce drastically the computation efficiency of the algorithm. On the other hand, a scaling factor s will be introduced to control the large values of the matrix multiplication. The SSA mechanism is defined as follows:

$$SSA^{'}(Q,K,V) = SN(QK^TV * s) \tag{15}$$

$$SSA(Q,K,V) = SN(BN(Linear(SSA^{'}(Q,K,V)))) \tag{16}$$

This implementation gets rid of the softmax layer, which was not a natural approximation of how the neurons activate.

IV. Conclusions

This work presents the core components of the Spikformers, highlighting the biological inspiration and the sequential data processing as the most relevant matters. Regarding biological modelling, action potential simulation model is described using the LIF Neuron which is used to deploy SNN architectures. As for dealing with sequential data, Transformers and Self-Attention methods are described and performance improvements are justify. After that, Spikformers are introduced as the intersection between SNNs and Transformers.

Experiments using static and neuromorphic datasets described in [3], reveal that the Spikeformer get closer results to the use of the transformer but with less computational resources and using x10 times less energy. Also, the Spikeformer outperforms other state-of-the-art SNN models.

References

[1] A. Turing, "Computer machinery and intelligence," *Mind*, pp. 433–460, Oct. 1950. DOI: 10.1093.
[2] A. Vaswani, N. Shazeer, N. Parmar, *et al.*, "Attention is all you need," Jun. 2017. DOI: 10.48550.
[3] Z. Zhou, Y. Zhu, C. He, *et al.*, "Spikformer: When spiking neural network meets transformer," Sep. 2022. DOI: 10.48550.

Isabel Prieto Payo was born in Huelva, Spain. She received the B.Sc. degrees in Telecommunications Engineering from the Universidad Politécnica de Madrid (UPM), Spain, in 2022. Since 2022 she is a M.Sc in Signal and Communication Theory student at Universidad Politécnica de Madrid (UPM), Spain.

Víctor Gutiérrez García was born in Barcelona, Spain. He received the B.Sc. degrees in Telecommunications Engineering from the Universidad Politécnica de Madrid (UPM), Spain, in 2022. Since 2023 he is a M.Sc in Signal and Communication Theory student at Universidad Politécnica de Madrid (UPM), Spain.