SpikingBERT: Distilling BERT to Train Spiking Language Models Using Implicit Differentiation

Malyaban Bal, Abhronil Sengupta
School of Electrical Engineering and Computer Science
The Pennsylvania State University
University Park, PA 16802
mjb7906@psu.edu, sengupta@psu.edu

Abstract

Large language Models (LLMs), though growing exceedingly powerful, comprises of orders of magnitude less neurons and synapses than the human brain. However, it requires significantly more power/energy to operate. In this work, we propose a novel bio-inspired spiking language model (LM) which aims to reduce the computational cost of conventional LMs by drawing motivation from the synaptic information flow in the brain. In this paper, we demonstrate a framework that leverages the average spiking rate of neurons at equilibrium to train a neuromorphic spiking LM using implicit differentiation technique, thereby overcoming the non-differentiability problem of spiking neural network (SNN) based algorithms without using any type of surrogate gradient. The steady-state convergence of the spiking neurons also allows us to design a spiking attention mechanism, which is critical in developing a scalable spiking LM. Moreover, the convergence of average spiking rate of neurons at equilibrium is utilized to develop a novel ANN-SNN knowledge distillation based technique wherein we use a pre-trained BERT model as "teacher" to train our "student" spiking architecture. While the primary architecture proposed in this paper is motivated by BERT, the technique can be potentially extended to different kinds of LLMs. Our work is the first one to demonstrate the performance of an operational spiking LM architecture on multiple different tasks in the GLUE benchmark. Our implementation source code is available at https://github.com/NeuroCompLab-psu/SpikingBERT.

Introduction

Large language Models (LLMs) are becoming increasingly popular because of its broad applications in a variety of natural language processing (NLP) tasks. LLMs like GPT-3 (Brown et al. 2020) has shown additional characteristics such as emergent abilities (Wei et al. 2022) which can only be realized once the model size/compute increases above a certain threshold. Recent times have witnessed commercial deployment of LLMs enabling worldwide reach and positively impacting real-world users. However, the immense power of LLMs comes at the cost of huge energy consumption both during the computationally expensive training phase as well as the inference phase. LLMs are characterized by large number of trainable parameters and are usually very deep. In order to alleviate the operational complexity of LLMs, we aim to draw motivation from the brain. Integrating the mechanism and knowledge embodied in LLMs into brain-inspired neural models hold immense promise for creating a bio-plausible and energy-efficient solution.

Spiking neural networks (SNNs) (Ghosh-Dastidar and Adeli 2009) are biologically inspired and communication between two neurons in an SNN architecture occurs in the form of spikes. This sparse spike-based information flow enables event-driven computation and communication in neuromorphic hardware, thereby resulting in significant energy savings (Sengupta et al. 2019). SNN based architectures have been also tested extensively on neuromorphic hardware like Intel's Loihi 2 processor (Davies et al. 2021) and have demonstrated orders of magnitude energy efficiency.

Scaling SNNs to complex domains, like NLP, poses significant challenges mainly due to the absence of scalable and efficient learning algorithms. The complexity of the tasks, coupled with the growing depth of the required model architectures, renders the practical application of BPTT infeasible. In this work, backed by robust theoretical foundations and empirical evidence, we explore a scalable framework for training spiking LMs. We consider our spiking LM as a dynamical system that, given an input, progressively converges to a steady-state (over T_{conv} time steps). Similar to most supervised learning algorithms, training is done in two phases, viz. "forward" and "backward". However, instead of learning through unrolling the computational graph over the operated time steps (like in BPTT), we leverage the convergence of the average spiking rate (ASR) of the neurons to an equilibrium state during the "forward" phase of learning. Upon convergence, we can derive fixed-point equations from the underlying model and subsequently employ implicit differentiation on the attained steady-state to train the model parameters effectively as described later on.

Training using implicit differentiation is primarily used in deep equilibrium models (DEQ) (Bai, Kolter, and Koltun 2019). Recently, this method has also been used for training of convolution based spiking architectures (Xiao et al. 2021) for vision related tasks. This methodology offers exceptional memory efficiency during training unlike BPTT, which requires a huge amount of memory to store a large computational graph. It also eliminates the necessity of surrogate gradient methods by implicitly calculating gradients, thereby circumventing the non-differentiability problem of spiking models. Under certain constraints (Bai, Kolter, and Koltun 2019), this form of learning is similar to bio-plausible and energy-based training methods like equilibrium propagation (Scellier and Bengio 2017; Bal and Sengupta 2022), thus bolstering a neuromorphic viewpoint of learning.

In transformer (Vaswani et al. 2017) based LMs as discussed in this paper, the attention mechanism serves as a vital component. However, vanilla attention mechanism is fundamentally non-spiking in nature, as it relies on sequences of real-valued vectors for the Query, Key, and Value components. In this paper, we present a spiking attention mechanism that utilizes spike-based inputs and operates over the number of time steps (T_{conv}) required for model convergence. The convergence of ASR of the neurons at equilibrium allows us to draw a close equivalence between the ASR of the spiking attention layer and vanilla attention.

Training LMs from scratch is a significant time and resource-intensive process. The additional overheads of training a spiking LM from scratch prompted us to seek out more proficient approaches for training our model. Knowledge distillation (KD) (Hinton, Vinyals, and Dean 2015) allows for faster and efficient transfer of knowledge from a trained "teacher" model to a possibly smaller in size "student" model. In this paper, we leverage the steady state ASR of the spiking LM and propose a novel ANN-SNN KD framework involving the ASR at equilibrium of specific intermediate layers of the "student" model and the activation of target layers of a larger pre-trained "teacher" model. Moreover, the feasibility of model training during distillation is enabled by the previously discussed training method. The primary contributions of our work are as follows:

- **SpikingBERT with Spiking Attention**: We propose a fully-operational spiking LM, following the architecture of BERT (Devlin et al. 2018), and evaluate it against different tasks (classification and regression) of the GLUE benchmark (Wang et al. 2018). We also propose an efficient spiking attention mechanism whose ASR at equilibrium approximates vanilla non-spiking attention.
- **Spiking LM Training:** We theoretically and empirically verify the convergence of our proposed spiking LM (comprising both linear and non-linear operations) to equilibrium state and use an implicit differentiation based method to overcome the non-differentiability issue of SNN training and reduce memory usage during training. This method enables training of Spiking LMs that surpass the scale of existing spiking models, thereby allowing development of deeper models for complex tasks.
- **ANN-SNN KD using Equilibrium States:** We leverage the equilibrium state of the neurons after convergence, to train our model more effectively using a novel ANN-SNN KD framework. This allows for developing an efficient and smaller spiking "student" model using larger BERT models as its "teacher".

Related Works

Spiking Architectures: Spiking architectures (Sengupta et al. 2019; Lee et al. 2020; Xiao et al. 2021; Zhou et al. 2022) have primarily been explored for vision based datasets such as CIFAR-100 (Krizhevsky, Nair, and Hinton 2009), ImageNet (Deng et al. 2009), among others as well as neuromorphic datasets such as NMNIST (Orchard et al. 2015), DVS Gesture (Amir et al. 2017). However, limited work has been done for sequence based NLP tasks. While most of the networks are non-attention based shallow models (Alawad, Yoon, and Tourassi 2017), recently (Zhu, Zhao, and Eshraghian 2023) explored developing a GPT-like model using linear attention. While GPT (Radford et al. 2018) is a decoder-only architecture, BERT is an encoder-only architecture whose ability to capture bi-directional contextual information makes it more suitable for text classification problems. Unlike our SpikingBERT, the SpikeGPT model uses surrogate gradient to overcome the non-differentiability problem and uses a BPTT approach for training. Furthermore, SpikingBERT is the first spiking LM, to the best of our knowledge, that has been evaluated against multiple different tasks in the GLUE benchmark. Moreover, due to the efficient KD incorporated in our approach, we can enhance model performance without the need for an extensive number of parameters. There has been some work on KD in SNNs previously, however all of them primarily explored BPTT based methods and focused solely on simple vision based datasets (Xu et al. 2023; Takuya, Zhang, and Nakashima 2021; Hong et al. 2023).

Efficient LMs: Given the increasingly growing scale of LMs, research focusing on attempting to make them computationally less expensive and smaller in size have gained significant attention. TinyBERT (Jiao et al. 2019) proposed extracting knowledge from the "teacher" model - from both the intermediate layers and the prediction layer. NAS-BERT (Xu et al. 2021) does model compression using neural architecture search. Work has also been done on distilling knowledge from BERT to a single layer BiLSTM network (Tang et al. 2019). Moreover, research endeavours have been made in methods like quantization (Kim et al. 2021), pruning (Kurtic et al. 2022), etc. to reduce the model complexity. Because appropriate neuromorphic baselines are currently unavailable, we compare our proposed model with existing standard NLP models and efficient LMs.

Methods

In this section, we will begin by examining the foundational principles of our method. Subsequently, we will delve into the architectural details to provide a comprehensive understanding. We delve into the theoretical and empirical foundations underlying the convergence of ASR in SNNs trained using implicit differentiation. Furthermore, we present an innovative approach to harness this convergence for the design of a spiking attention mechanism and a novel KD mechanism leveraging a pre-trained BERT model as a "teacher", thereby enhancing the learning process. We also elaborate on the framework for using implicit differentiation based technique to train our spiking architecture.

Spiking Neural Networks

The fundamental building block of the proposed spiking architecture comprises of leaky integrate and fire (LIF) neu-

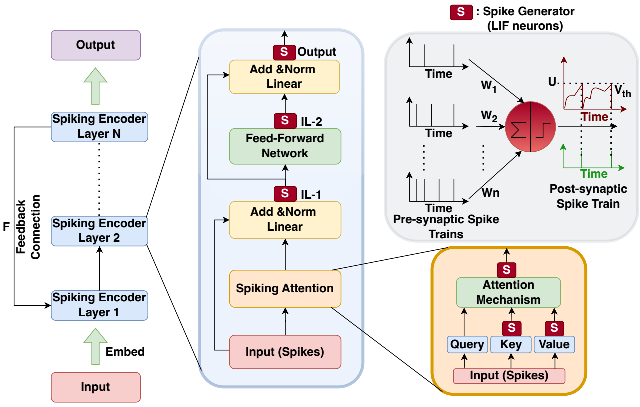

Figure 1: High-level overview of the SpikingBERT model. During the "forward" phase of learning, the network is simulated over T_{conv} time steps, i.e., until the ASR of the neurons of each layer converges to an equilibrium. Information flow both within and between two spiking encoders occur using spikes instead of real values, thereby mimicking event-driven information flow in bio-inspired systems.

rons. The internal dynamics of a simple LIF neuron is similar to a biological neuron and is given as follows,

$$\tau_m \cdot \frac{du}{dt} = -(u(t) - u_{rest}) + R \cdot I(t) \tag{1}$$

where, u is the membrane potential; u_{rest} is the resting membrane potential; $I(t)$ is the input voltage at time t scaled by a constant, R, representing resistance; τ_m is the time constant. Moreover, if $u > V_{th}$, then the neuron potential is updated by subtracting V_{th} and the neuron emits a spike, i.e., it sends out a '1' else '0'. A suitable discrete time representation of the dynamics for the i^{th} neuron, can be described as follows,

$$\begin{aligned} u_i[t+\delta] &= \gamma u_i[t] + \sum_j (w_{ij} s_j[t]) + b_i, \\ s_i[t+1] &= S(u_i[t+\delta]), \\ u_i[t+1] &= u_i[t+\delta] - V_{th} s_i[t+1] \end{aligned} \tag{2}$$

where, γ is the leaky term related to the constant τ_m in Eqn. 1 (for LIF neurons, we keep $\gamma < 1$ and for IF $\gamma = 1$); s_j is the spike from the j^{th} input neuron; w_{ij} is the synaptic weight of the connection between the pre-synaptic and post-synaptic neurons; $t + \delta$ represents an intermediate time step representation to determine if the neuron has fired; b_i represents a bias term; S is the non-differentiable function for spike generation and subtraction is used as reset operation.

Implicit Modeling

Implicit modeling takes a different approach by not explicitly defining the precise computation of a model's output from its input. Instead, it relies on imposing specific constraints on the model, ensuring that these constraints are met to achieve the desired results. For example, consider a simple model represented by a function h. In order to formulate an explicit model with input $x \in X$ and output $z \in Z$, the following computation is performed: $z = h(x)$. However, for formulating it implicitly, a function $g : X \times Z \to R^n$ is defined, such that $g(x, z) = h(x) - z$ and the goal will be to find the root of the equation: $g(x, z) = 0$. While this simple example demonstrates algebraic equations, these methodologies can be extended to fixed-point equations, thereby paving the way for the development of DEQ (Bai, Kolter, and Koltun 2019).

Let us consider a fixed-point equation of the form $z = f_\theta(z)$, where θ is the set of parameters. This fixed point equation converges over time, i.e., after T_{conv} time steps $z_{T_{conv}} = z_{(T_{conv}+1)}$, thereby reaching an equilibrium state. Similarly, as before, we can form another equation namely, $g_\theta(z) = f_\theta(z) - z$. Here, the loss function L that we will be defining will utilize the value of z at equilibrium, i.e., $z_{T\,conv} = z^*$. Using implicit differentiation (Bai, Kolter, and Koltun 2019), the following relation can be derived,

$$\frac{\partial L(z^*)}{\partial \theta} = -\frac{\partial L(z^*)}{\partial z^*} (J_{g_\theta}^{-1}|_{z^*}) \frac{\partial f_\theta(z^*)}{\partial \theta} \tag{3}$$

where, $J_{g_\theta}^{-1}|_{z^*}$ is the inverse Jacobian of g_θ when $z = z^*$, i.e., at equilibrium. The proposed spiking LM architecture follows a similar set of equations which will be described in the following section. Since the gradient is computed using implicit differentiation on the converged steady-state, we avoid the non-differentiability issues of the spiking function (Neftci, Mostafa, and Zenke 2019). Furthermore, by computing gradients solely at the equilibrium state, there is no requirement to store intermediate hidden states. This characteristic enhances the scalability and memory efficiency of this approach in comparison to BPTT.

Architecture

The high-level building block of the proposed spiking LM comprises of Spiking Encoder (SE) layers, which can be considered similar to individual encoder layers in a transformer architecture. Both intra and inter-layer communication in the SE layers occur using spikes at every time step during the "forward" phase and spiking LIF neurons are fundamental units in its design. As shown in Fig. 1, each SE layer consists of a Spiking Attention layer followed by fully connected layers (some including skip-connections similar to those in BERT) viz. Intermediate Layer-1 (IL-1), Intermediate Layer-2 (IL-2) and an output layer, all of which operate using spikes. The structure of the proposed network comprises of N stacked SE layers similar to the structure of BERT. The input embeddings are processed by an LIF neuron layer, generating spikes that are propagated through the model. In some of the internal layers of SE, we use layer normalization (following BERT).

From a biological perspective, feedback connections are present in the human-brain and moreover in some cases (Kubilius et al. 2019) shallower network with recurrent connections shows performance comparable or better than deeper architectures. The connection (F) is added from the output of the final SE layer to the first one in order to introduce a feedback. The feedback connection is an optional component that adds to the model's bio-plausibility. The general formulations of steady-state ASR equations, developed subsequently, can be seamlessly applied to models involving both feedback connections and those without any feedback. Unlike in vision-based tasks, feedback did not improve performance considerably when compared with no-feedback scenario in our experiments with GLUE benchmark. However, we still explore it on a theoretical level to maintain consistency with previous works (Xiao et al. 2021) and to encourage future research on feedback enabled SNNs in other domains.

Similar to vanilla transformer based architectures, the input sequence is directly fed into the model. The model converges over T_{conv} time steps during the "forward" phase to settle to an equilibrium state. As discussed earlier, the spiking neurons of the model have their individual membrane potentials, u, which are updated at every time step during convergence. At time $t+1$, the membrane potential of u_1, i.e., the input to the first SE layer can be given as,

$$\begin{aligned} u_1[t+1] = \gamma u_1[t] + Fs_{(N,out)}[t] \\ +W_0(x) + b_1 - V_{th}s_1[t+1] \end{aligned} \tag{4}$$

where, W_0 provides the embedding of the input sequence x of length N_s and produces a sequence of vectors $y \in R^{N_s \times D_{emb}}$ (D_{emb} is the encoding dimension), F is the weight of the feedback connection (if feedback is included), b_1 is bias and $s_{(N,out)}[t]$ are spikes generated from the N^{th} SE layer in the previous time step. The membrane potential of a layer $i > 1$ can be represented simplistically as,

$$\begin{aligned} u_i[t+1] = \gamma u_i[t] + W_{(i-1)}(s_{(i-1)}[t+1]) + b_i \\ -V_{th}s_i[t+1] \end{aligned} \tag{5}$$

where, $W_{(i-1)}$ is an operation (as formulated by each individual layer) defined on a set of spikes from previous layers. The described LIF neurons propagate information using spikes that are generated following Eqn. 2.

The average spiking rate (ASR) of a neuron at layer i can be defined as, $a_i[t] = \frac{\sum_{\tau=1}^{t} \gamma^{t-\tau} s_i[\tau]}{\sum_{\tau=1}^{t} \gamma^{t-\tau}}$. Given $W_{(i-1)}$ is a linear operation, using Eqn. 5 and performing a weighted (if $\gamma < 1$) average over time (with $u[0] = 0, s[0] = 0$) we get,

$$a_i[t+1] = \frac{1}{V_{th}}(W_{(i-1)}a_{(i-1)}[t+1] + b_i - \frac{u_i[t+1]}{\sum_{i=0}^{t} \gamma^i}) \tag{6}$$

Since, $a_i[t]$ represents ASR, its value is restricted within [0,1]. Following previous work on implicit differentiation at equilibrium (Xiao et al. 2021), as the average of input converges to equilibrium $\bar{x}[t] \rightarrow x^*$, then the ASR of the layers (Eqn. 6) in the spiking architecture converges to equilibrium points: $a_i[t] \rightarrow a_i^*$ (with bounded random error in case of LIF neurons). At equilibrium, the ASR a_i^* of layer i satisfies,

$$a_i^* = \sigma(\frac{1}{V_{th}}(W_{(i-1)}(a_{i-1}^*) + b_i)) \tag{7}$$

where clipping function $\sigma(x)$ bounds the values within [0,1].

Like the linear operations, the layers with non-linear operations such as spiking attention also converges to a steady-state ASR as is empirically validated in this paper (Fig. 2a).

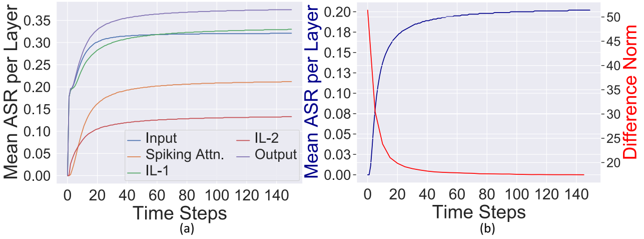

Figure 2: Results obtained after passing a randomly sampled input from SST-2 dataset through SpikingBERT$_4$. (a) Graph showing mean (over number of neurons) of the ASR of different sub-layers in an SE layer against the operating time steps. (b) The y-axis on the left depicts mean (over number of neurons) of the ASR of a randomly chosen spiking attention layer. Along the right y-axis, the "Difference Norm" between the output of the steady-state equation of the chosen spiking attention layer and the calculated ASR is shown. Time steps used for convergence in shown along the x-axis.

Steady-state equations like Eqn. 7 are leveraged during training. Thus, for the spiking attention layer, we formulate a surrogate steady state equation at equilibrium given as,

$$a_{(attn)}^* = \sigma(\frac{1}{V_{th}}(Attn(a_x^*, a_k^*, a_v^*) + b_{(attn)}) \tag{8}$$

where, a_x^* is the ASR of the layer used to form Query, a_k^* is the ASR of the Key and a_v^* is the ASR of the Value. Operational details of the spiking attention mechanism and empirical convergence and justification of the defined equation is discussed in the next subsection.

Thus, after the model converges to equilibrium, the dynamics of the steady-state ASR of the underlying SNN can be mapped to a surrogate non-spiking architecture where the input and output of each layer are the corresponding ASRs (a_i^*). The operation of individual layers in the surrogate network is given by the steady-state equations as described earlier and can be simplified to the form, $a_i^* = l_i(a_j^*, \dots)$ where, l_is are steady-state equations corresponding to each layer like Eqn. 7 and 8. The parameters ($a_j^*, \dots$) associated with each layer (l_i) are defined according to the specific operation. If we use feedback connection, the fixed-point equation of the first layer is of the form $a_1^* = l_1(l_M \circ \dots \circ l_2(a_1^*), x^*)$ where, $l_1(a, x) = \sigma(\frac{1}{V_{th}}(Fa + W_0(x) + b_1))$ with M being the total number of individual layers.

For the task of text classification, we use the last layer of the network, i.e., the output of the N^{th} encoder layer given as $a_{N,out}^*$ as an input to a linear classification function. Simulating the network for T_{conv} time steps, we can compute $a_{N,out}[T] = \frac{\sum_t (s_{N,out}[t])}{T}$ (for simplicity of demonstration, $\gamma = 1$), which we can use as $a_{N,out}^*$. Moreover, since the behaviour at equilibrium is captured by the surrogate network using only ASR, we can simply perform backpropagation to train the weights by leveraging Eqn. 3. Thus, instead of performing BPTT to train the underlying spiking architecture, we use simple backpropagation to train the weights of the spiking LM using only equilibrium state ASR of neurons.

Spiking Attention Mechanism

We propose a computationally efficient Spiking Attention mechanism where the inputs are processed as spikes from the previous layer. The proposed attention operations of the module at time step t can be formulated as,

$$Attn(S_x(t), S_K(t), S_V(t)) = \\ \pi(s * Q(S_x(t))(S_K(t))^T) \cdot S_V(t) \tag{9}$$

where, $Q(S_x(t))$ is obtained after passing input spikes ($S_x(t)$) at time t through a linear layer (W_Q) for generating Query. The spikes corresponding to the Key layer ($S_K(t)$) is computed by passing the input spikes $S_x(t)$ through a linear mapping (W_K), connected to an LIF neuron layer, as illustrated in Fig. 1. $S_V(t)$ is obtained similarly using linear mapping (W_V). π is usually the softmax function and s is a scaling factor where generally $s = \frac{1}{\sqrt{d_k}}$, with d_k being the encoding dimension of Key. Recent work has shown that the non-linear normalization operation π is not always essential (Zhou et al. 2022). The operations outlined in Eqn. 9 exhibit characteristics akin to spiking architectures. This is because performing the aforementioned matrix multiplications entails multiplying a real-valued matrix with a matrix composed of spikes (i.e., '0's and '1's) at each step. Thus, instead of requiring $O(n^3)$ floating point multiplicative and $O(n^3)$ accumulative operations, we can implement the attention mechanism utilizing only $O(n^3)$ accumulative operations - which has been shown to significantly reduce computation cost in SNNs (Sengupta et al. 2019) (note that this is a first order estimate ignoring memory transactions). The output of this module is passed through an LIF neuron, resulting in spikes that are fed to the next layer (Fig. 1).

The empirical convergence of the ASR of attention layer is demonstrated in Fig. 2b. As discussed earlier, we construct a surrogate steady-state function at equilibrium which helps us in efficient training of the model. The empirical rationale for employing this specific functional form is substantiated by observing the reduction in the difference norm between the output of the surrogate equation and the computed ASR of the layer at each timestep, as demonstrated in Fig. 2b. Thus, using Eqn. 8 and considering no bias, $V_{th} = 1$ and no clamping function σ, we see that as the model converges in time, the actual ASR of the spiking attention layer $a_{attn}[t]$ approximates vanilla attention given by $Attn(a_x[t], a_k[t], a_v[t])$.

ANN-SNN KD using Equilibrium States

The proposed architecture and training mechanism guarantee the steady-state convergence of ASR of neurons across all layers in the models (Fig. 2a), including the internal representational layers. This enables us to develop an ANN-SNN based KD framework, using the intermediate layer ASR at equilibrium and the activations of internal layers of the "teacher" BERT models. In order to make the learning faster and more efficient, we propose transferring knowledge from a pre-trained LM, such as BERT, to our spiking LM. KD is done over the internal layers such as transformer layers, embedding-layer as well as the prediction layer.

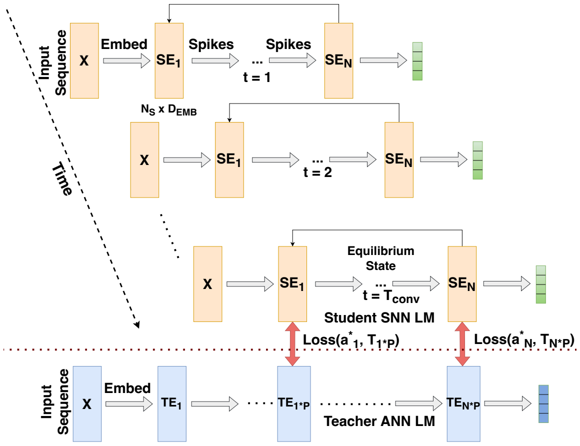

Figure 3: High-level overview of transformer layer based KD at equilibrium (following Eqn. 10) from a "teacher" LM to a spiking "student" LM.

To perform transformer layer-based distillation, we utilize the output layer of each SE layer (Fig. 3). Subsequently, we establish the loss function by comparing the ASR (at equilibrium) of the output from each SE layer with the activations of the corresponding mapped encoder layers in the "teacher" model. The loss function using Mean Squared Error (MSE) is formulated as follows,

$$\mathcal{L}_{h_i} = MSE(a^*_{h_i} W_{T_d}, T_{f(h_i)}) \tag{10}$$

where, $a^*_{h_i}$ is the ASR (at equilibrium) of the output neurons of the i^{th} SE layer in the "student" and $T_{f(h_i)}$ is the output of the $f(h_i)^{th}$ layer of the "teacher". W_{T_d} is a linear transformation that maps the "student" layer to the same dimension as the corresponding "teacher" network layer. Function f maps "student" layer h_i to a specific target layer in the "teacher" network. In our approach, we have used the following mapping function: $f(h_i) = h'_{p*i}$; $p = T_{enc}/S_{enc}$, where h'_i is the output of the i^{th} encoder layer of the "teacher" and T_{enc} and S_{enc} are the number of encoder layers in the "teacher" and "student" models respectively.

By leveraging the equilibrium state of the neurons, KD utilizes the converged ASR to its advantage. Consequently, we employ implicit differentiation technique (Eqn. 3) for training using the equilibrium state of the intermediate layers. This enables us to perform a faster and efficient layer-wise knowledge transfer from a pre-trained ANN-based LM to a smaller (in size) spiking LM. Moreover, each spiking encoder layer within the "student" model incorporates a spiking attention layer as described in Fig. 1. We strengthen knowledge transfer further by optimizing an MSE loss over the attention score at equilibrium of the "student" network with the attention score of the corresponding mapped attention layer of the "teacher" network (following the function f). We also perform embedding layer level distillation by formulating a loss function similar to Eqn. 10. We use ASR of the embedding layer (input to the first spiking encoder layer) and create an MSE loss against the embedding layer of the "teacher".

Model	**QQP**	**MNLI-m**	**SST-2**	**QNLI**	**RTE**	**MRPC**	**STS-B**
CBoW (Wang et al. 2018)	75.00	57.10	79.50	62.50	71.90	75.00/83.70	70.60/71.10
BiLSTM (Wang et al. 2018)	85.30	66.70	87.50	77.00	58.50	77.90/85.10	71.60/72.00
BiLSTM + Attn, CoVe (Wang et al. 2018)	83.50	67.90	89.20	72.50	58.10	72.80/82.40	59.40/58.00
GenSen (Wang et al. 2018)	82.60	71.40	87.20	62.50	78.40	80.40/86.20	81.30/81.80
BERT$_5$ + PF (Xu et al. 2021)	84.10	67.70	81.60	80.90	62.80	78.60/-	-/81.10
NAS-BERT$_5$ + PF (Xu et al. 2021)	85.70	74.20	84.90	83.90	67.00	80.00/-	-/82.80
NAS-BERT$_5$ + KD (Xu et al. 2021)	85.80	74.40	87.30	84.90	66.60	79.60/-	-/83.00
NAS-BERT$_{10}$ + PF (Xu et al. 2021)	88.40	76.00	88.60	86.30	68.70	81.50/-	-/84.30
BERT$_{\text{TINY}}$ Adam (Frantar, Kurtic, and Alistarh 2021)	81.09	65.36	80.11	77.85	-	69.90/-	64.39/-
BERT$_{\text{MINI}}$ Adam (Frantar, Kurtic, and Alistarh 2021)	86.45	73.30	85.46	83.85	-	76.57/-	82.09/-
SpikingBERT$_4$	**86.82**	**78.10**	**88.19**	**85.20**	**66.06**	**79.17/85.15**	**82.20/81.90**
TinyBERT$_4$ (no DA) (Jiao et al. 2019)	88.50	80.60	90.50	87.00	68.20	82.40/-	86.20/85.70

Table 1: Results showing performance of our model (SpikingBERT$_4$) against some standard models and other efficient implementations of BERT on GLUE evaluation set. Accuracy is used as the metric for QQP, MNLI-m, SST-2, QNLI, RTE datasets while both accuracy and F1 scores are reported for the MRPC dataset. For STS-B, we report Pearson/Spearman correlation.

Post-transformer layer distillation, we also perform prediction layer distillation, following the works of (Hinton, Vinyals, and Dean 2015). The loss at the prediction layer for classification tasks can be written as,

$$\mathcal{L}_{pred} = CE(c(a^*_{pred})/t^{'}, T_{pred}/t^{'}) \qquad (11)$$

where, a^*_{pred} is the ASR at equilibrium of the output of the final Spiking Encoder, c acts as a linear mapping and T_{pred} is the output logits of the "teacher" network. CE is the cross-entropy loss function and $t^{'}$ is temperature.

Distillation is done in two different stages following (Jiao et al. 2019). Firstly, we perform general distillation where we use a pre-trained general BERT model and use general domain data not specific to any particular task. Secondly, we perform task specific distillation on datasets relevant to the particular task using a task-specific fine-tuned BERT model as a "teacher". By employing a two-staged distillation process, we significantly enhance the efficiency of our spiking LM development, while also resulting in a substantially reduced "student" model size when compared to the "teacher".

Experimentation

In this section, we demonstrate the performance of our proposed spiking LM and evaluate it against different tasks in the General Language Understanding Evaluation (GLUE) benchmark (Wang et al. 2018). We also highlight the core hyper-parameters for training SpikingBERT model. We also perform extensive analysis to report energy and power efficiency of our proposed model. The experiments were run on Nvidia RTX A5000 GPUs (8) each with 24GB memory.

Datasets

In order to evaluate our model, we chose seven different type of tasks (**six classification and one regression task**) from the GLUE benchmark. We chose Quora Question Pair (QQP), Microsoft Research Paraphrase Corpus (MRPC) and Semantic Textual Similarity Benchmark (STS-B) (**regression** task) to evaluate our model on similarity and paraphrase tasks. For inference tasks, we opted for Multi-Genre Natural Language Inference (MNLI), Question-answering NLI (QNLI) and Recognizing Textual Entailment (RTE) datasets. For single-sentence based sentiment analysis tasks, we chose Stanford Sentiment Treebank (SST-2).

Baselines & SpikingBERT Settings

To the best our knowledge, our proposed model is the first one to report and analyze the performance of a spiking LM on different tasks from the GLUE benchmark. (Zhu, Zhao, and Eshraghian 2023) proposed a spiking based GPT architecture and it reported 80.39% accuracy on SST-2 dataset with a model (45M) comparable with our model size (50M). Using a larger model of size 216M, SpikeGPT achieves 88.76%, which is comparable to our performance. Our model is able to demonstrate higher accuracy with lower number of parameters primarily because of the KD technique used to train it as well as because of the BERT-based architecture which is suitable for classification problems due to its bidirectional context understanding. In addition to the above mentioned work, we focus primarily on comparing the performance of our architecture against existing non-spiking methodologies that aims to reduce the complexity of base-BERT model. However, unlike the proposed model, these methods are not applicable for a spiking implementation on neuromorphic chips such as Loihi 2 since all of them are non-spiking architectures. Our goal for the comparisons is to show that low-powered Spiking LM with less trainable parameters can achieve similar accuracy compared to other efficient LM implementations (number of parameters less than 50M). Moreover, since this is the first time spiking LMs have been evaluated against a benchmark, we did not use additional techniques such as data augmentation (Jiao et al. 2019), etc. to boost model performance in order to delineate the core advantages of our proposed training method.

For all the tasks, we keep the maximum sequence length at 128. The encoding dimension of the tokens in the input is 768 and the intermediate (IL-2) size of the model is 3072. In order to emphasize on the benefits of KD, SpikingBERT$_4$ comprises of only 4 SE blocks compared to 12 encoders blocks of BERT. Increasing the number of SE blocks will also improve the overall model performance. The model trained for reporting the results (Table 1) did not have a feedback, since adding it did not increase accuracy.

Hyper-parameters	Range	Optimal
T_{conv}: General KD	(5-150)	80
T_{conv}: Task-based IKD	(5-150)	80
V_{th} (Threshold Voltage)	(0.25 - 5.0)	1.0
γ (Leak term)	(0.8 - 1.0)	.99 (LIF); 1 (IF)
$t^{'}$ (Temperature)	(0.1 - 10.0)	1.0
Batch Size: General KD	(8-256)	128
Batch Size: Task-based IKD	(8-128)	[16,32]
Epochs: General KD	-	5
Epochs: Task-based IKD	-	20

Table 2: Hyper-parameters (explored range and optimal values) for SpikingBERT$_4$ used across all datasets.

During training of SpikingBERT, we perform general distillation first using English Wikipedia as text corpus and keeping the sequence length at 128. Transformer (& Embedding) layer distillation following Eqn. 10 is performed and the "teacher" is a pre-trained BERT$_{\text{BASE}}$ model (uncased). Following this, we perform task-based internal layer KD (IKD) with corresponding fine-tuned BERT models and perform it both on the inner transformer layers and the embedding layer. Core hyper-parameters associated are given in Table 2. We found that grouping IKD based on type of task (i.e., inference, similarity, etc.) at this stage improves performance. For example, once a task-specific distillation is done using MNLI dataset, if we use that distilled model (as "student") and then perform task-specific distillation on QNLI dataset, we achieve higher accuracy on QNLI dataset. After task-based IKD is done, we finally perform prediction-layer distillation following Eqn. 11 to develop the final model. It is to be noted that if we directly train our model on true labels at this stage, we obtain similar results in terms of accuracy. Without using the proposed KD, there is at least 4% to 5% drop in accuracy across all datasets.

Analysis of Power & Energy Efficiency

The proposed spiking LM consists of less number of parameters than the "teacher" BERT models (109M) and in addition to that it uses only accumulative (ACC) operations in place of multiplicative and accumulative operations (MAC) found in vanilla BERT models. Considering 45nm CMOS technology, ACC operations exhibit an impressive energy efficiency, consuming only 0.9pJ, which is over five times (5.1) more efficient than MAC operations that demand 4.6pJ (Han et al. 2015). For estimating the energy and power efficiency of our spiking LM, we leverage the concept of normalized operations (Lu and Sengupta 2020), which considers the spiking rates of each layer and corresponding layer-wise operations. The total normalized OPS can be defined as $Norm\#OPS = \frac{\sum_i IFR_i * Layer\#OPS_{i+1}}{\sum Layer\#OPS}$, where IFR_i is the total number of spikes over inference time steps averaged over number of neurons. Thus, energy-efficiency factor of an SNN, which can be given by the ratio of energy consumed by an iso-architecture ANN over the proposed SNN can be expressed as: $e = (\frac{1}{5.1} * Norm\#OPS)^{-1}$. SNNs operate on specific time steps, allowing them to dynamically balance accuracy and energy consumption. We perform an extensive energy-accuracy tradeoff analysis on the SST-2 dataset. After conducting general and task-based IKD, during the final training phase, we train a set of models with different values of T_{conv} to see its effect on energy consumption and accuracy. The energy-efficiency factor (e) and the obtained accuracy w.r.t the time steps (T_{conv}) is demonstrated in Fig. 4a. A suitable tradeoff point can be found at $T_{conv} = 16$, where we achieve an accuracy close to the highest value (2% difference) but we are able to achieve nearly twice energy-efficiency than a non-spiking model of same size.

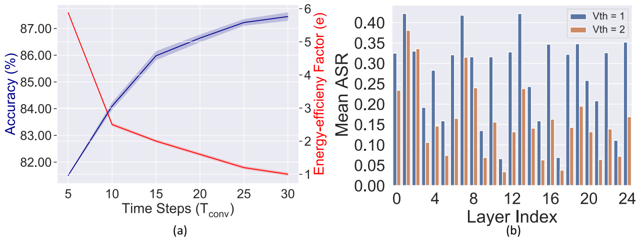

Figure 4: Results obtained on SST-2 dataset. (a) Variation of Accuracy and Energy-efficiency factor (e) as T_{conv} increases. (b) Variation in mean ASR per neuron in different sub-layers of SpikingBERT$_4$ following changes in V_{th}.

Moreover, by increasing V_{th}, we can reduce the ASR of neurons at each layer, leading to a significant decrease in power consumption. We perform an ablation study on the effects of V_{th} on ASR, which is reported in Fig. 4b. Increasing V_{th} intuitively also increases the convergence time steps, thus making energy-consumption effectively similar. However, it allows us to reduce the instantaneous power-consumption considerably - ideal for edge computing.

Conclusion and Future Works

Drawing inspiration from the astonishing intricacy of the human brain, a complexity that outshines that of any current LLM, we have the opportunity to leverage these insights in crafting models that not only replicate biologically plausible behavior but also offer energy-efficient solutions through minimal power consumption. In this paper, we propose a spiking LM and evaluate it against multiple tasks in the GLUE benchmark. Leveraging steady-state convergence, we introduced a spiking attention mechanism, proposed a novel ANN-SNN based KD for faster and efficient learning and explored training of Spiking LMs using implicit differentiation, thereby overcoming multiple issues affecting training of SNN models. Implementing our model on neuromorphic hardware such as Loihi 2 for inference will help us develop a low-powered solution which can potentially be implemented on edge devices.

Further endeavours can be made to extend this methodology to design other spiking LMs such as GPT, etc. There is still a performance gap between the proposed spiking LM and BERT-based fine-tuned models. We can work towards closing this gap by delving into diverse spiking neuron models, examining temporal encoding schemes, and incorporating graded spikes, among other strategies.

Acknowledgments

The work was supported by the National Science Foundation CAREER Award under Grant #2337646 and by Oracle Cloud credits and related resources provided by the Oracle for Research program.

References

Alawad, M.; Yoon, H.-J.; and Tourassi, G. 2017. Energy efficient stochastic-based deep spiking neural networks for sparse datasets. In *2017 IEEE International Conference on Big Data (Big Data)*, 311–318. IEEE.

Amir, A.; Taba, B.; Berg, D.; Melano, T.; McKinstry, J.; Di Nolfo, C.; Nayak, T.; Andreopoulos, A.; Garreau, G.; Mendoza, M.; et al. 2017. A low power, fully event-based gesture recognition system. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 7243–7252.

Bai, S.; Kolter, J. Z.; and Koltun, V. 2019. Deep equilibrium models. *Advances in Neural Information Processing Systems*, 32.

Bal, M.; and Sengupta, A. 2022. Sequence Learning using Equilibrium Propagation. *arXiv preprint arXiv:2209.09626*.

Brown, T.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J. D.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33: 1877–1901.

Davies, M.; Wild, A.; Orchard, G.; Sandamirskaya, Y.; Guerra, G. A. F.; Joshi, P.; Plank, P.; and Risbud, S. R. 2021. Advancing neuromorphic computing with loihi: A survey of results and outlook. *Proceedings of the IEEE*, 109(5): 911–934.

Deng, J.; Dong, W.; Socher, R.; Li, L.-J.; Li, K.; and Fei-Fei, L. 2009. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, 248–255. Ieee.

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Frantar, E.; Kurtic, E.; and Alistarh, D. 2021. M-FAC: Efficient matrix-free approximations of second-order information. *Advances in Neural Information Processing Systems*, 34: 14873–14886.

Ghosh-Dastidar, S.; and Adeli, H. 2009. Spiking neural networks. *International journal of neural systems*, 19(04): 295–308.

Han, S.; Pool, J.; Tran, J.; and Dally, W. J. 2015. Learning both Weights and Connections for Efficient Neural Networks. arXiv:1506.02626.

Hinton, G.; Vinyals, O.; and Dean, J. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Hong, D.; Shen, J.; Qi, Y.; and Wang, Y. 2023. LaSNN: Layer-wise ANN-to-SNN Distillation for Effective and Efficient Training in Deep Spiking Neural Networks. *arXiv preprint arXiv:2304.09101*.

Jiao, X.; Yin, Y.; Shang, L.; Jiang, X.; Chen, X.; Li, L.; Wang, F.; and Liu, Q. 2019. Tinybert: Distilling bert for natural language understanding. *arXiv preprint arXiv:1909.10351*.

Kim, S.; Gholami, A.; Yao, Z.; Mahoney, M. W.; and Keutzer, K. 2021. I-bert: Integer-only bert quantization. In *International conference on machine learning*, 5506–5518. PMLR.

Krizhevsky, A.; Nair, V.; and Hinton, G. 2009. Cifar-10 and cifar-100 datasets. *URl: https://www. cs. toronto. edu/kriz/-cifar. html*, 6(1): 1.

Kubilius, J.; Schrimpf, M.; Kar, K.; Rajalingham, R.; Hong, H.; Majaj, N.; Issa, E.; Bashivan, P.; Prescott-Roy, J.; Schmidt, K.; et al. 2019. Brain-like object recognition with high-performing shallow recurrent ANNs. *Advances in neural information processing systems*, 32.

Kurtic, E.; Campos, D.; Nguyen, T.; Frantar, E.; Kurtz, M.; Fineran, B.; Goin, M.; and Alistarh, D. 2022. The optimal bert surgeon: Scalable and accurate second-order pruning for large language models. *arXiv preprint arXiv:2203.07259*.

Lee, C.; Sarwar, S. S.; Panda, P.; Srinivasan, G.; and Roy, K. 2020. Enabling spike-based backpropagation for training deep neural network architectures. *Frontiers in neuroscience*, 119.

Lu, S.; and Sengupta, A. 2020. Exploring the connection between binary and spiking neural networks. *Frontiers in neuroscience*, 14: 535.

Neftci, E. O.; Mostafa, H.; and Zenke, F. 2019. Surrogate gradient learning in spiking neural networks: Bringing the power of gradient-based optimization to spiking neural networks. *IEEE Signal Processing Magazine*, 36(6): 51–63.

Orchard, G.; Jayawant, A.; Cohen, G. K.; and Thakor, N. 2015. Converting static image datasets to spiking neuromorphic datasets using saccades. *Frontiers in neuroscience*, 9: 437.

Radford, A.; Narasimhan, K.; Salimans, T.; Sutskever, I.; et al. 2018. Improving language understanding by generative pre-training.

Scellier, B.; and Bengio, Y. 2017. Equilibrium propagation: Bridging the gap between energy-based models and backpropagation. *Frontiers in computational neuroscience*, 11: 24.

Sengupta, A.; Ye, Y.; Wang, R.; Liu, C.; and Roy, K. 2019. Going deeper in spiking neural networks: VGG and residual architectures. *Frontiers in neuroscience*, 13: 95.

Takuya, S.; Zhang, R.; and Nakashima, Y. 2021. Training low-latency spiking neural network through knowledge distillation. In *2021 IEEE Symposium in Low-Power and High-Speed Chips (COOL CHIPS)*, 1–3. IEEE.

Tang, R.; Lu, Y.; Liu, L.; Mou, L.; Vechtomova, O.; and Lin, J. 2019. Distilling task-specific knowledge from bert into simple neural networks. *arXiv preprint arXiv:1903.12136*.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, L.; and Polosukhin, I. 2017. Attention Is All You Need. arXiv:1706.03762.

Wang, A.; Singh, A.; Michael, J.; Hill, F.; Levy, O.; and Bowman, S. R. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*.

Wei, J.; Tay, Y.; Bommasani, R.; Raffel, C.; Zoph, B.; Borgeaud, S.; Yogatama, D.; Bosma, M.; Zhou, D.; Metzler, D.; Chi, E. H.; Hashimoto, T.; Vinyals, O.; Liang, P.; Dean, J.; and Fedus, W. 2022. Emergent Abilities of Large Language Models. arXiv:2206.07682.

Xiao, M.; Meng, Q.; Zhang, Z.; Wang, Y.; and Lin, Z. 2021. Training feedback spiking neural networks by implicit differentiation on the equilibrium state. *Advances in Neural Information Processing Systems*, 34: 14516–14528.

Xu, J.; Tan, X.; Luo, R.; Song, K.; Li, J.; Qin, T.; and Liu, T.-Y. 2021. NAS-BERT: task-agnostic and adaptive-size BERT compression with neural architecture search. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, 1933–1943.

Xu, Q.; Li, Y.; Shen, J.; Liu, J. K.; Tang, H.; and Pan, G. 2023. Constructing deep spiking neural networks from artificial neural networks with knowledge distillation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 7886–7895.

Zhou, Z.; Zhu, Y.; He, C.; Wang, Y.; Yan, S.; Tian, Y.; and Yuan, L. 2022. Spikformer: When spiking neural network meets transformer. *arXiv preprint arXiv:2209.15425*.

Zhu, R.-J.; Zhao, Q.; and Eshraghian, J. K. 2023. Spikegpt: Generative pre-trained language model with spiking neural networks. *arXiv preprint arXiv:2302.13939*.