Biologically Plausible Learning for NLP Using Spiking Neural Network

TAZRIAN HOSSAIN[1], ADIBA AMREEN ALAM[2], NAFISA RAHMAN[3], MAISA KABIR[4], NAFIS AL SHAMS[5], DR. MD. GOLAM RABIUL ALAM[6]

[1]BRAC University, Dhaka, Bangladesh (e-mail: tazrian.hossain08@gmail.com)
[2]BRAC University, Dhaka, Bangladesh (e-mail: adibaamreenalam@gmail.com)
[3]BRAC University, Dhaka, Bangladesh (e-mail: nafisa.rahman750@gmail.com)
[4]BRAC University, Dhaka, Bangladesh (e-mail: maisa.kabir2001@gmail.com)
[5]BRAC University, Dhaka, Bangladesh (e-mail: nafisdrubo@gmail.com)
[6]BRAC University, Dhaka, Bangladesh (e-mail: rabiul.alam@bracu.ac.bd)

ABSTRACT Commonly referred to as the third generation of neural networks, Spiking Neural Networks (SNNs) have attracted plenty of research interest in the last decade mainly due to its energy efficient and biologically realistic approach. Although areas like computer vision and signal processing have benefited significantly from SNNs, it seems NLP is still uncharted territory in neuromorphic devices. Our research seeks to establish the capability of Spike-Timing-Dependent Plasticity (R-STDP) in SNNs to conduct sentiment analysis. R-STDP provides a reward based learning mechanism that adjusts the synaptic weights according to the spike timing and feedback such as classification accuracy. This duplicates dopamine-controlled learning in the human brain. We also employed the Forward-Forward algorithm which replaces traditional backpropagation with local, layer-wise learning based on positive and negative sample contrast, allowing for modular and decentralized training without the need for backward error signals which further enhances biological plausibility. In addition, we employ an optimized rate coding method to convert textual data into spike trains that can then be easily processed by SNN architectures. We show that by applying this model on a benchmark sentiment analysis and affective computing dataset, SNNs, using learning rules such as R-STDP, can harness energy efficiency and the event-based nature of neuromorphic platforms to achieve sentiment classification accuracy (48%, 73%) comparable to conventional approaches. To understand the findings of our work, we compare our model to the existing deep learning models. The results obtained are of particular interest in order to assess the performance of spiking models for low-power NLP tasks, and to tailor SNNs into further machine learning pipelines.

INDEX TERMS Affective Computing; Dopamine; Forward-Forward; Natural Language Processing; NLP; R-STDP; Spiking Neural Networks; Sentiment Classification; SNN; Synaptic Plasticity; STDP

I. INTRODUCTION

Natural Language Processing (NLP) is a field of Artificial Intelligence that is focused on the interaction between computers and human beings with the use of natural language. It consists of algorithms and models which allow a machine to learn, understand, and respond to humans. NLP has a wide area of applications using different kinds of neural networks like Artificial Neural Networks (ANN), Convolutional Neural Networks (CNN) and Spiking Neural Networks (SNN). ANNs and CNNs rely on backpropagation - a learning method that, despite its effectiveness, is widely considered biologically implausible due to its reliance on global error signals, symmetric weight transport, and sequential updates, none of which are observed in the human brain [35].

SNNs, on the other hand, closely mimic the way biological brains function. SNN uses binary signals and inherently processes time-based data, making them ideal for neuromorphic computing, robotics or sensory data processing [29]. The two primary fields of SNN research at the moment are computer vision and information processing where it has achieved a performance comparable to real-valued equivalent neural networks (although with a slight performance reduction). Furthermore, the energy consumption of SNN is far lower than that of their real-valued equivalents [36]. This is why it is intriguing to examine popular NLP tasks in this context of neuromorphic computing since many research subjects, including NLP, are currently understudied [2] in the setting of spiking neural networks.

In this research, we explore the use of SNNs for NLP tasks by first employing rate coding [8] to convert word embeddings into spike trains, making linguistic data compatible with spiking models. We then utilize Reward-Modulated

Spike-Timing-Dependent Plasticity (R-STDP), a biologically inspired learning rule that adjusts synaptic weights based on both spike timing [30] and external reward signals [32]—similar to dopaminergic modulation in biological brains. However, to further increase the biological plausibility and efficiency of learning, we integrate Forward-Forward (FF) learning [37] into our architecture. FF training replaces backward error propagation with two forward passes: one with positive (correct) data and one with negative (incorrect) data. Each layer learns locally to increase its activation for positive inputs and decrease it for negatives, aligning closely with how distributed and parallel learning occurs in the brain.

By combining R-STDP with the FF algorithm, our proposed hybrid framework not only avoids biologically implausible mechanisms but also enables efficient, reward-sensitive and modular learning for NLP tasks, pushing the boundary of neuromorphic computing into a largely unexplored yet promising domain.

A. RESEARCH PROBLEM

NLP has been an appealing field of research for an extended period of time now. Consequently, the advancements in this field are highly notable. However, existing research in this field mostly focused on and prioritized the accuracy of the models and accepted the high energy consumption as a necessary trade-off. Thus, the resource-consuming and power-hungry process desperately needs optimization in regards to its energy consumption [38].

This is where SNNs fit into the picture. As SNNs are event-driven in nature, during most of its life-cycle, the neurons remain dormant until activated by the spiking mechanism. This seemed like a suitable approach to the problem of energy optimization. However, in the case of SNN, the trade-off between accuracy and energy consumption still maintains a gap and without bridging this gap, researchers cannot obtain the same performance they have come to expect from ANNs and conventional models [39]. Therefore, researchers shifted their focus to optimize the accuracy of the SNN models without significant energy consumption trade-off. Different researchers have come up with different ways to optimize the SNN models. Some researchers proposed better binarization and encoding methods [8]. On the other hand, researchers also focused on the neural processing of the models itself. Since SNN exclusively uses discrete inputs and generates discrete spikes, conventional training methods like backpropagation are poorly suited for SNNs. Several solutions to this have been proposed and they have variable results. There are mainly 2 different research paths that have been widely used. One is using a surrogate gradient to back propagate and reduce errors [3]. Another path is using Spike Time Dependent Plasticity (STDP), an unsupervised process that strengthens and weakens neural connections [4]. In order to avoid the additional computations of a surrogate gradient, some researchers have focused on optimizing the STDP itself. One of those ways to improve on STDP is R-STDP (Reward Modulated) which has already been used in several researches [5], [6], [12].

Despite their potential, three underexplored opportunities hinder progress. Firstly, most R-STDP implementations use basic rate-coding or ANN-to-SNN conversion techniques [45], neglecting advanced spike-encoding methods that could enhance both accuracy and energy efficiency. Moreover, backpropagation's incompatibility with neuromorphic systems [46] contrasts with emerging alternatives like the forward-forward algorithm, which mimics cortical layer-wise learning. However, this method has not been systematically adapted for SNNs in NLP. Also no studies combine R-STDP with optimized rate-coding and biologically plausible learning rules (e.g., forward-forward) to create end-to-end efficient SNN architectures for language tasks.

B. RESEARCH OBJECTIVE

This research aims to advance energy-efficient and biologically plausible NLP by exploring an under-investigated combination of Spiking Neural Networks, Reward-Modulated STDP, and backpropagation-free learning. The main objectives of the research is given below

- To optimize energy consumption of NLP tasks (Sentiment Analysis and Affective Computing) using SNNs. Existing NLP tasks are processed by Transformer based Artificial Neural Networks and LLM models which take a significant number of operations to complete and are inefficient in terms of energy consumption.
- To enhance the learning capability of SNNs for NLP tasks. Existing SNN models compromise performance in order to conserve energy. In the present state, SNN models can't effectively compete with other models in terms of performance.
- To provide biological plausibility to sentiment classification and affective computation tasks. SNN models, by nature, are more close to a biological brain than other existing models.
- To analyze and compare the performance to other models and combinations with different rate-coding techniques. SNN models require spike-trains as inputs and the process of generating these spike-trains can have a notable effect on the model's performance. Comparison of models and their rate-coding techniques is a core objective of our research.
- To contribute practical insights and recommendations for future research on building biologically inspired, energy-aware models for language processing.

II. LITERATURE REVIEW

A. EXISTING WORKS

1) Biologically Inspired SNNs

Spiking Neural Networks (SNNs) offer a more biological and realistic approach as an alternative to traditional artificial neural networks due to the nature of using spike-based

communication, membrane potential modeling, and temporal dynamics. SNN uses various neuron models, among which Leaky Integrate-and-Fire (LIF) helps to naturally generate spikes (discrete events) [31] which are more relevant in the field of SNN. LIF has a lot of advantages such as biological plausibility, compatibility with neuromorphic hardwares, scalability in large networks, and anaytical tractability. However, there are some challenges while implementing LIF such as the non-differentiable activation function. To overcome such problems, different approaches can be used. One of the approaches is surrogate gradient descent [40]. Surrogate gradients provide an approximate way to compute gradients during the backpropagation process in SNN. Instead of using the non-differentiable spiking function, a smooth surrogate function is introduced, which allows computation of gradients while the original function remains the same. This approach enables gradient-based optimization, preserves temporal information while also maintaining the spiking dynamics, and also allows a much more efficient path for training deep SNN. The SNN networks align with different cognitive processes that are observed in the human brain such as excitatory and inhibitory signaling, sparse activation, and local plasticity [9]. Models such as excitatory-inhibitory cooperative iterative learning (EICIL) have emulated neural behaviors by maintaining a biologically grounded ratio of 80:20 between the excitatory and inhibitory neurons [10]. On the other hand, other biologically inspired works include dendritic spine-based synaptic pruning strategies [11] and adaptive threshold mechanisms like Bio-inspired Dynamic Energy-Temporal Threshold (BDETT), which stabilizes the firing rate in a dynamic manner that changes the environments [12].

Consequently, SNN's biological significance makes it a perfect place to use brain activities that are inspired by biology in a variety of computational domains. In addition to the methods discussed above, the biologically inspired SNN rule that will mainly concern our research is that of synaptic plasticity.

2) SNNs for NLP tasks

The traditional NLP relies on dense continuous embeddings, which are not compatible with the discrete and temporal inputs that are required by SNNs. [8] proposed a deterministic spike encoding scheme and their model achieved 32x inference and 60x training efficiency compared to ANNs. Another model, SpikeGPT [42] , replaced the multi-head-self-attention mechanism with a sequential attention process. It achieved competitive accuracy while reducing operations by 32x on neuromorphic hardware. It became one of the largest backpropagation-trained SNN models to date.

However, it needs to be noted that both SNNLP and SpikeGPT focused on energy efficiency rather than strict biological plausibility. Therefore, exploring the use of biologically inspired methods such as R-STDP and forward-forward learning for NLP tasks still remains an intriguing field of study.

B. BACKGROUND

1) Spike Train encoding

In natural Language Processing, the initial salient part of the entire process is to encode the input text into a continuous n-dimensional feature vector termed as 'embedding' [13] which maps each token in a multidimensional vector space. Extensive research has been done already in this field to encode input data into embedding vectors. Many word embedding techniques have already been proposed. For example, FastText [14], Word2Vec [15], GloVe [16], etc. In addition, research has not been only limited to Word Embedding. NLP has been a field of extensive research for years and recent research works are capable of encoding larger language aspects like sentences [17], [18] and phrases [14] while retaining the syntactic information of the original input.

In conventional NLP methods, embedding itself is enough. However, due to SNN's discrete nature of input, these embedded feature vectors cannot be processed using spiking architecture as it requires a spike train input. To solve this problem, the embedded vectors are binarized and the vectors are converted into vectors that only consist of 0s and 1s. There are 2 widely used methods to obtain these binarized vectors. They are:

a: Binarization Methods:

There are several proposed methods for binarization like the 'Near-lossless' [19] binarization method. However, most of these methods use an autoencoder to convert embedded vectors into vectors of 0/1. Which, by itself, is not that energy consuming considering it only needs to be done once. However, when it comes to Sentences, binarization requires use of Transformer [20] models. In such a case, the energy consumption of a model using both autoencoders and Transformer models is not ideal for a paper like this since the entire point of using the Spiking Neural Structure was to optimize energy consumption.

b: Rate-Coding:

In this method, floating-point numbers are easily converted to a series of spikes based on a Poisson process, where, for a number of time-steps k:

$$Y_i = \begin{cases} 1 & \text{if } X_i(t) < p_i \\ 0 & \text{if } X_i(t) \geq p_i \end{cases} \tag{1}$$

Given x $\sim$ Binomial (k,p) This process is repeated for all items in the input vector v, generating a spike train for each of the input vector's feature dimensions [8].

For the purpose of this paper, we looked at several different rate-coding methods until [8] proposed its own rate-coding technique that was appropriate for our energy optimization

task. Some rate-coding techniques as well as [8]'s proposed "SNN-rate" has been briefly discussed below:

- ANN - This is a typical ANN using regular floating-point embeddings
- SNN-rate-rand - This is a Poisson rate-coded embedding being used in a downstream SNN
- SNN-bin - This is a binarized embedding being used in a downstream SNN
- SNN-rate [8]: Poisson rate-coded embedding method that generates deterministic spike trains.

c: SNN-rate:

As this is the encoding system we will use for this paper, we need more details on how it works. This method aims to inherently modify the stochastic Poisson rate-coding method (referred to as "SNN-rate-rand") which generates spike trains that approximate the float-point input values stochastically and instead generates deterministic spike trains for the given embedding. The paper [8] generates deterministic encoding as follows:

$$V_i(t) = V_i(t-1) + x_i(t) \tag{2}$$

$$Y_i(t) = \begin{cases} 1 & \text{if } V_i(t) \geq \alpha_i \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

Where x_i is the floating-point value being encoded, $V_i(t)$ is the membrane potential of the neuron at time-step t, and α_i is the membrane threshold of the neuron.

This deterministic encoding, used in downstream SNNs, ensures higher fidelity spike trains, leading to improved model performance compared to traditional rate-coding methods. The overall goal is to maintain high accuracy while leveraging SNN's energy efficiency benefits in comparison to conventional neural network models.

2) Spike-Timing-Dependent Plasticity (STDP) and its Variants

STDP is an algorithm where the time difference between pre- and post-synaptic neurons determine the strengthening or weakening of a synapse. Traditionally, the equation of STDP is as follows:

$$\Delta t = t_{\text{post}} - t_{\text{pre}}, \tag{2.4}$$

$$W(\Delta t) = \begin{cases} A^+ e^{-\Delta t/\tau^+}, & \text{if } \Delta t \geq 0, \\ -A^- e^{\Delta t/\tau^-}, & \text{if } \Delta t < 0, \end{cases} \tag{2.5}$$

$$\Delta w = \sum_{t_{\text{pre}}} \sum_{t_{\text{post}}} W(\Delta t). \tag{2.6}$$

Here, w is the synaptic weight. Δw is the change of the synaptic weight. t_{pre} and t_{post} stand for the timing of the firing spike from pre-neuron and post-neuron. A+ is the magnitude of the weight change when we have a positive correlation, i.e. the pre-synaptic neuron fires prior to the post-synaptic neuron and A is the magnitude when the correlation is negative. τ+ and τ are time constants defining the width of the positive and negative learning window. [27]

A simple image classification problem which used spike coding to represent images and STDP to update weights, found that their model offered 20 times faster computing speed while retaining accuracy when compared to traditional spiking neuron models [21]. One of the most common ways for training artificial neural networks (ANNs) is the backpropagation algorithm but it does not resemble how humans learn and requires a large amount of labeled data. On the other hand, despite reducing computational and memory complexity, SNNs trained solely with STDP tend to be sub-optimal due to local optimization and neuronal coordination issues. Research exists on a number of adaptive mechanisms to improve STDP such as introduction of a dynamic firing threshold, removal of noisy inputs, lateral inhibition and STB-STDP which batches both samples and time-steps. Although the accuracy achieved by this model on the CIFAR10 dataset was relatively low compared to supervised learning algorithms, it was remarkable for a solely unsupervised method [22]. Another method, called sym-STDP, utilizes the concept of symmetric STDP where synaptic changes occur regardless of the order of the firing neuron. It is a biologically plausible, high-performance supervised learning framework that uses spike-based learning rules and achieved high accuracy on the MNIST dataset [23]. Therefore, biologically plausible mechanisms that improve STDP are valuable assets for both supervised and unsupervised learning.

In an effort to modify STDP, a reinforcement learning rule, R-STDP was introduced which was inspired by the role of neuromodulators such as Dopamine (DA) or Acetylcholine (Ach) in the modulation of STDP. Dopamine increases the plasticity of the eligible synapses. All eligible synapses are either working against obtaining the reward, toward it, or the activity is unrelated to the reward. If they systematically work against the behavior being rewarded, STDP will weaken these synapses. If they work toward the goal, the effect of STDP will be to strengthen them. Any uncorrelated activity will lead to a very small change in synaptic strength [28]. [7] created a 3 layer DCSNN where each convolutional layer was followed by a pooling layer. For every decision made, it was compared with the original label. Accordingly, a reward or punishment signal was generated that would be passed back to the layers that implemented R-STDP. This study achieved a high accuracy on the MNIST dataset and also predicted that R-STDP can be useful when the dataset is populated with frequent distractors. Another study [5], utilized R-STDP in medical image classification to address the issue of class imbalance. Higher rewards were provided to the minority class whereas lower rewards were provided to the majority class which significantly improved the sensitivity of disease identification. Additionally, many hierarchical SNNs with R-STDP learning rules have been used to solve object recognition tasks and achieved notable results [24], [25]. Hence,

while the dopamine mechanism has been extensively used across various computer vision tasks, SNN itself remains significantly underused in the domain of NLP. [26] For the task of sentiment classification, a simple reward signal can be computed which rewards correct classification and punishes wrong classification. The weight of the synapses can then be updated by R-STDP accordingly. 1 illustrates how R-STDP affects learning in spiking neural networks. Part (a) shows the timing of spikes from pre- and post-synaptic neurons.Part (b) shows the eligibility trace, which increases or decreases based on spike timing and decays over time. Part (c) shows external reward or punishment signals and part (d) shows changes in synaptic weight, which increase after a reward and decrease after a punishment. Overall, the image demonstrates how spike timing combined with reward signals drives synaptic plasticity and learning.

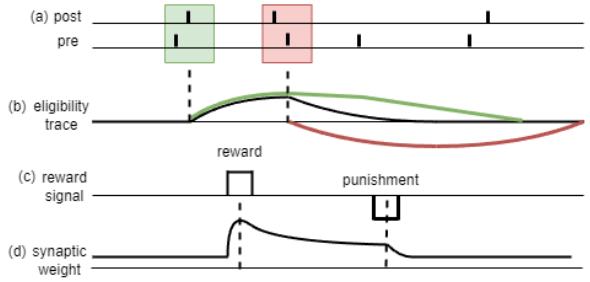

FIGURE 1. How R-stdp affects learning

3) The Forward-Forward Algorithm

The FF algorithm was introduced by [37] , which is a biologically inspired training paradigm that eliminates the use of backpropagation. Instead of using traditional backward gradient propagation techniques, it employs two forward passes, one with real (positive) and the other with corrupted (negative) data. Each of the layers updates the weight based on a local system of "goodness" points, typically a function that uses the sum of squared activations.

$$G^{(l)} = \sum_i \left(a_i^{(l)}\right)^2 \tag{4}$$

Here, $a_i^{(l)}$ denotes the activation of the i-th neuron in layer l. This metric serves as a local, unsupervised signal that quantifies how strongly the layer responds to the input.

$$\Delta W^{(l)} \propto G_{\text{pos}}^{(l)} - G_{\text{neg}}^{(l)} \tag{5}$$

Where $G_{\text{pos}}^{(l)}$ and $G_{\text{neg}}^{(l)}$ represent the goodness scores from the positive and negative inputs, respectively. This formulation allows each layer to locally assess whether its representation favors the real data over the corrupted counterpart, enabling learning without the need for backpropagation.

[41] extended FF algorithm to NLP, specifically for sentiment analysis on the IMDb dataset. They trained the FF networks without backpropagation using Word2Vec embeddings and embedded label representations. They introduced a scheme called pyramidal threshold optimization scheme, where deeper layers had higher loss thresholds, simulating the hierarchical abstraction of the human brain. This network then achieved a comparable performance to backpropagation-trained models with an improvement of up to 11% in accuracy under optimized threshold scheduling. Their study also proved that unbounded activations like ReLU were more compatible for the FF algorithm than sigmoid or tanh.

Based on the foundation built by [37], [**?**] adapted the FF Framework for the spiking neural architectures by using LIF neurons and spike count-based activations. They calculated the "goodness" from the squared spike count of each neuron across 2 forward passes. They also introduced surrogate gradient approximations and deterministic spike encoding, which allowed the model to train effectively without the use of traditional backpropagation. The FF-SNN was tested upon various datasets like MNIST, Fashion-MNIST, CIFAR-10, SHD and N-MNIST which demonstrated that the accuracy was competitive with conventional backpropagation methods. Its compatibility with neuromorphic hardware and local weight update rules makes the FF-SNNs more suitable for low-powered AI systems.

C. KEY FINDINGS

Both the methods, R-STDP and FF Algorithm offer a biologically plausible alternative to gradient backpropagation while targeting different facets of neural computation. R-STDP mainly focuses on the synaptic plasticity modulated using reward signals and excels in dynamic learning scenarios, like in vision. On the other hand, FF provides a structured, gradient-free framework that can be used for both ANN and SNN settings while achieving competitive accuracy with energy-efficient operations. When these two methods are combined with novel spike-train encoding strategies, they present promising paths for the development of brain-like neural architectures that are capable of learning from both images and texts without having to sacrifice computational efficiency.

III. METHODOLOGY

A. TOP LEVEL OVERVIEW

In order to optimize sentiment analysis and affective computing and compare the results with existing data, we first have to implement the proposed model itself.

Figure 2 illustrates the complete operational pipeline of a biologically inspired SNN model designed for sentiment and emotion classification in NLP. The system replaces traditional Poisson and label encoders with a deterministic spike-based input encoding mechanism (SNN-rate), converting GloVe embeddings into spike trains. It uses decentralized layer-wise training using R-STDP and FF learning paradigm. Two model clones, *alice* and *alice2*, independently train the

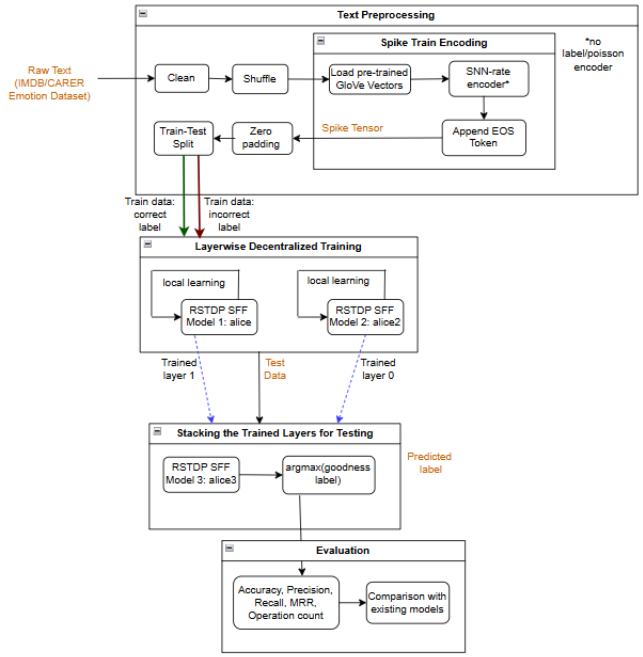

FIGURE 2. Top Level Overview of the Proposed System

second and first hidden layers, respectively. Their learned weights are then merged into *alice3* for final inference using spike-based "goodness" scores. The model's performance is assessed using accuracy, MRR, precision, recall, and energy efficiency, and benchmarked against conventional ANN and SNN baselines.

Our plan has 5 distinct stages:

1) **Dataset Preprocessing**: At this stage, the input data is preprocessed to make it suitable for our analysis. This part is mainly focused on formatting the data and features.
2) **Processing of Input**: Most NLP tasks require further processing of the data. In addition, due to using a SNN model, the input needs to be further processed into discrete values instead of floating-points/actual values. Therefore, this particular stage of processing the data can be divided into several sub-stages:
   - **Splitting Dataset**: This involves separating the portion of the dataset the model will be trained on and the portion that will be used for testing and prediction.
   - **Embedding**: Converting both training and testing datasets into feature vectors.
   - **Rate-Coding**: Converting the embedded vectors into vectors consisting of discrete values (0/1), which can now be considered a spike train. For this part, we are using [8]'s custom 'SNN-rate' for rate-coding. This will be done for both training and testing datasets.
3) **Training**: In this stage, we implement the proposed reinforcement learning technique (R-STDP) to train the model on the training dataset.
4) **Prediction**: This stage focuses on using the proposed model to predict the classifications of the test dataset that we split earlier.
5) **Analysis and Comparison**: Finally, we analyze and compare the obtained results (e.g., accuracy, energy consumption) with different models that performed the same task.

B. DATASETS

We used two benchmark datasets for the purpose of sentiment and affective computation analysis.

IMDB Dataset: It is a binary sentiment classification dataset with 50,000 entries. Given a string of text from a movie review, the target is to decide whether the text is expressing positive or negative sentiment [33].

CARER Dataset: It is a multi-class emotion dataset with 20,000 entries. Given a tweet as text, the target is to decide which of the 6 emotions- sadness, joy, love, anger, fear, or surprise, are being depicted in the text [34].

C. DATA PREPROCESSING

In order to test our proposed SNN model, it is vital that we use datasets that are widely available and frequently used for benchmark testing. The datasets themselves have been explained in the Dataset section. However, in order to use these benchmark datasets, data preparation is necessary. The entire process of data preparation/ data preprocessing is explained below.

The data loading mechanism is invoked by the training module. In this stage of the preprocessing, the datasets library is used to load the pre-existing dataset. A custom clean-text function is used to change the text into lowercase, remove underscores, whitespaces and punctuations. The dataset is then randomly shuffled to ensure random distribution of output labels.

At this stage, the training module calls on the dataset pre-processing method which does 5 things:

- Initializes pre-trained GloVe embeddings.
- Since our model is spike event driven, just embeddings are not usable as inputs. In this regard, we use a slightly altered version of SNN-rate, the optimized deterministic rate-coding mechanism that converts these embeddings into a spike train. For more information on SNN-rate and how it works, we can refer to [8].
- Each token in the sentence is embedded into a binary vector. Bit positions with 1s indicate active neurons. These binary vectors are then encoded as spike trains over time using a configurable repetition factor and spike frequency.
- A spike train is constructed for each input sample with shape $[T, \text{embed\_dim}]$, where each word contributes to

a specific time slice. An End-Of-Sequence (EOS) token is appended to ensure a fixed-length temporal window.
- All samples are padded to a uniform temporal dimension (512 for IMDB, 32 for CARER) and returned as `torch.Tensor` objects ready for spiking network training.

The spike trains are then split into 70% train, and 30% test sets.

TABLE 1. Dataset Split Distribution

Dataset	Train	Test	Total
IMDB	35,000	15,000	50,000
CARER	14,000	6,000	20,000

This split dataset is finally returned back to the training module where it goes through further processing.

D. PROPOSED SNN MODEL ARCHITECTURE (SFF-RSTDP)

The architecture proposed in our work is a biologically-inspired SNN built on the SFF paradigm. It replaces traditional backpropagation with a decentralized, layer-wise R-STDP learning mechanism, along with surrogate loss calculation. This approach enables parallel and independent training of layers and aligns with the sparse, energy-efficient principles of neuromorphic computing. The structure of our model and its flow of operation is explained below.

1) Model Structure

The main spiking network, termed as *NetofLain* is a fully connected network where each layer is a spiking neural layer based on a custom class called *LayerofLain*. The overall architecture, shown in figure 3, has 3 layers. Our model architecture was inspired and influenced by [43]

- **Input Layer**: Accepts time-sequenced spike trains of shape [T, batch size, embed dimension], where each token in a sentence contributes to a portion of the temporal spike train. The dimension of this layer changes depending on the embedded dimension of the input spike trains.
- **Two hidden layers**: Each layer is defined as a fully connected linear spiking layer (*spikingjelly.layer.Linear*) followed by an Integrate-and-Fire neuron (*spikingjelly.neuron.IFNode*) with surrogate gradients (ATan). After fine tuning, each layer has a dimension of 500.
- **Output logits**: Since classification is made by calculating aggregating spike-based "goodness" across time and layers, our model does not have an output layer like conventional neural network models.

2) Parameters

For the parameters, we set the decay for pre-synaptic activity to 2.0. The decay for post-synaptic activity is 100.0. The learning rate is 0.003 for the first layer and 0.004 for the second layer. Both the positive and negative reward threshold are 0.005. Maximum timestep is 512 for IMDB and 32 for CARER emotion dataset. Lastly, the batch size is set to 15.

3) Decentralized structure

Our proposed model of network clones the same model twice. Each clone termed as *alice* and *alice2* has the same structure of [input dimension x 500 x 500]. In decentralized learning mechanism, *alice* separately trains only its second layer, using inputs that pass through the first layer. Similarly, *alice2* only trains its first layer using the raw spike encoded inputs. Finally, the learned weights from both the models are stacked into a new model called *alice3* which is an object of *NetofLain* class. This final *alice3* model is used to compute goodness scores across the test data and finally make predictions. Figure 4 shows an overview of how each model learns in a decentralized manner.

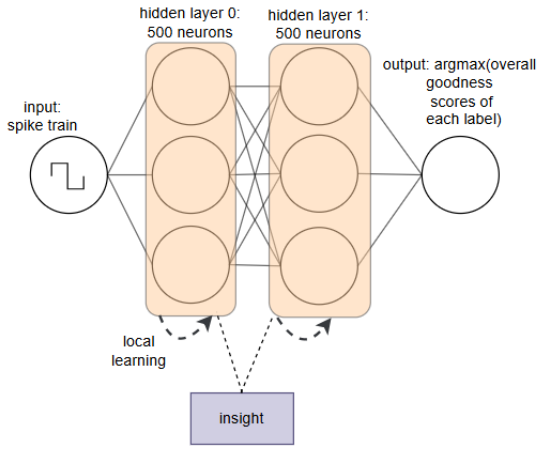

FIGURE 3. SFF-RSTDP Model (Note that the network doesn't output a class directly as it computes goodness per class label and chooses the class with maximum goodness)

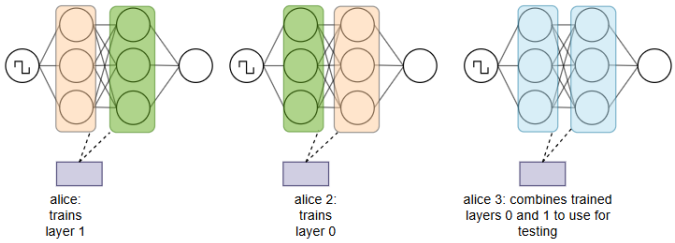

FIGURE 4. Decentralized Layer-wise Training Strategy

4) Forward-Forward Mechanism

To implement the biologically plausible Forward-Forward training paradigm, instead of back propagation, our model corrects itself by forward passing positive and negative samples.

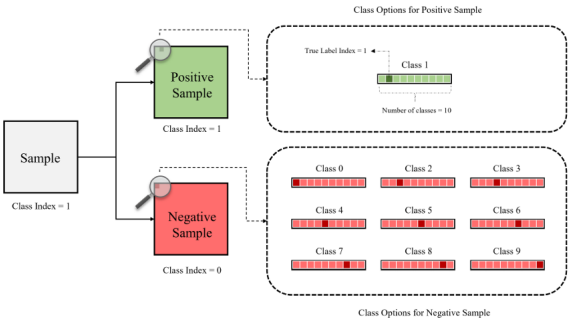

FIGURE 5. Different options for overlaying a label on a sample of a 10-class dataset for positive and negative sample generation [35]

First, it creates 2 tensors which are respectively positive and negative feature-label samples. An example of positive and negative sample generation for a 10 class dataset is given by figure 5. These positive and negative samples are passed through the network (forward pass without learning) to produce corresponding spikes. From the obtained output, the goodness and negative goodness of each layer for this specific sample is calculated and the sum of mean goodness of all the layers for each sample is recorded.

The goodness is calculated by:

$$G = \frac{1}{T}\sum_{t=1}^{T}\left(\frac{1}{N}\sum_{i=1}^{N} x_i(t)^2\right) \tag{6}$$

Where G is the goodness score of the layer for one input sample, T is the total number of time steps, N is the number of neurons in the layer, $x_i(t)$ is the membrane potential of neuron i at time step t.

Finally, once the goodness measure for the sample is calculated and tracked, our model utilizes the goodness to calculate weight update for that layer only. A high goodness indicates that the layer is confidently excited by the input (positive sample) while a low goodness indicates that the layer is not excited (negative sample).

5) Weight Update

In the second forward pass, each layer updates its weights using a combination of Reward-Modulated Spike-Timing-Dependent Plasticity (R-STDP) and a surrogate gradient-based loss function shown by figure 6. The mechanism of each is given below.

Reward-Modulated STDP (R-STDP): In conventional STDP, if a presynaptic neuron fires shortly before a post-synaptic neuron, the synaptic connection is strengthened; otherwise, it is weakened. In R-STDP, this plasticity rule is modulated by a scalar reward signal.

In our proposed model, the reward is derived from the layer's *goodness*, which is calculated during the initial (non-learning) forward pass. The reward is applied at each time step during the second forward pass and determines the magnitude and direction of weight updates. It is defined as:

$$\text{Reward}_{\text{pos}} = \alpha_{\text{pos}} \cdot (G - \theta_{\text{pos}}) \tag{7}$$

$$\text{Reward}_{\text{neg}} = \alpha_{\text{neg}} \cdot (\theta_{\text{neg}} - G) \tag{8}$$

Where:

- G: goodness score of the current layer
- $\theta_{\text{pos}}, \theta_{\text{neg}}$: threshold parameters for positive and negative samples
- $\alpha_{\text{pos}}, \alpha_{\text{neg}}$: reward scaling factors

If the layer fires strongly for the wrong input ($R < 0$), synapses are punished (weakened). If it fires strongly for the correct input ($R > 0$), synapses are reinforced (strengthened). These updates are handled by the *MSTDPLearner* object from the `SpikingJelly` library.

Surrogate Gradient Loss: To complement the STDP-based updates, a surrogate gradient-based loss is also used to optimize weights via gradient descent. This loss ensures separation between the positive and negative views by comparing their combined goodness values.

A collaborative signal, called *insight*, is computed as the sum of the goodness from all other layers (excluding the one currently being trained). For a layer i, the combined scores are:

$$\text{Combined}_{\text{pos}} = G_i^+ + \sum_{j \neq i} G_j^+ \quad \text{(insight\_pos)} \tag{9}$$

$$\text{Combined}_{\text{neg}} = G_i^- + \sum_{j \neq i} G_j^- \quad \text{(insight\_neg)} \tag{10}$$

- G_i^+: Goodness score of the current layer i for a positive (correctly labeled) input.
- G_i^-: Goodness score of the current layer i for a negative (incorrectly labeled) input.
- G_j^+: Goodness score of layer $j \neq i$ for the same positive input.
- G_j^-: Goodness score of layer $j \neq i$ for the same negative input.

The final loss is calculated using a smooth log-sum-exp function that promotes contrast between the two views.

$$\mathcal{L}_{\text{mixed}} = \log\left(\exp(\text{Combined}_{\text{pos}}) + \exp(\text{Combined}_{\text{neg}}) + 1\right) \tag{11}$$

Where L is the surrogate loss. This loss allows for gradient-based fine-tuning using optimizers like Adam, with gradient flow enabled by surrogate activation functions (e.g., ATan).

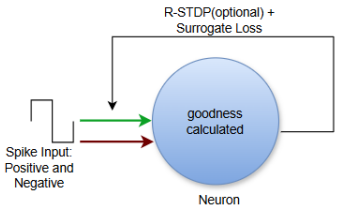

FIGURE 6. Neuron-level learning loop: Each neuron receives spike input from both positive and negative views. It computes its activation goodness (based on membrane potential), then adjusts synaptic weights using R-STDP and surrogate loss to favor correct labels and suppress incorrect ones.

IV. EXPERIMENTAL RESULTS

In order to achieve the objectives of this research paper, some metrics and goals had to be set for proper evaluation of the results. The purpose of our model, when broken down, ultimately strives to achieve the following:

1) Improvement Over Existing SNN Models: Enhance the performance of SNN models that rely solely on gradient-based backpropagation. Spiking Neural Networks with non-deterministic rate-encoding were the baseline for this metric. In the result section, these models are referred to as “SNN” and "SNN-rate".
2) Efficiency: Significantly reduce the total number of operations required to train a model that achieves acceptable levels of accuracy. Existing ANN models were the baseline for this metric. In the result section, this is referred to as “ANN”.
3) Optimization Without Gradient Backpropagation: Outperform or match the performance of SNN-rate encoded optimized SNN models while avoiding the complexities and memory constraints associated with gradient-based backpropagation. SNN models optimized through gradient backpropagation and custom rate-encoding technique, as referenced in [8] was considered to be the baseline for this metric. In the result section, this is referred to as “SNN-rate-op”.
4) Research Gap Reduction: Contribute to the research on using Reward Modulated STDP (Spike-Timing-Dependent Plasticity)and Forward-Forward mechanism on SNN models for NLP tasks such as sentiment analysis and addressing the current lack of research in this area.

For benchmark testing, the number of epochs for all models except our proposed model were set to 10. All 5 models (ANN, SNN, SNN-rate (Non-deterministic SNN), SNN-rate-op (Deterministic SNN) and SFF-RSTDP [this paper’s proposed model] were trained on 2 datasets, IMDB and CARER emotion dataset. For comparison, 5 metrics were focused on: Accuracy, mean reciprocal rank (MRR), precision, recall, and total number of operations. How these metrics were calculated have already been explained in details.

A. TRAINING MODULE

The training module is the core of our system where everything is sequentially called on and executed. The entire flow of operation of the training module is explained below:

The training module can be separated into 5 operations:

1) Calls on the Dataloader to retrieve the appropriate dataset (IMDB or CARER) and split the dataset as required (Explained in Dataset section). Finally return the training and validation data.
2) Invokes the SNN-rate encoding mechanism and obtains the optimized rate-encoded embeddings which will be used as the input for training the Network.
3) Sets the amount of neurons in the first layer of the SNN model based on the size of the input embedding.
4) Training Loop: Loops through the training process for 1 epoch.
5) The training loop runs independently for each clone of the network (*alice* and *alice2*). For each input, the model creates positive and negative samples by combining the same input spike with the correct and incorrect labels. Both samples are passed through the network in a forward pass (without learning). From the output spikes, the model calculates the goodness for each layer. The average goodness from the preceding layers is also recorded and returned as "insight". Reward is computed and based on this reward, the model applies R-STDP updates (if enabled) and computes surrogate loss to update the synaptic weights locally for the active layer. The first clone, *alice*, trains the second hidden layer while keeping the first layer frozen. The second clone, *alice2*, trains the first layer using raw input spikes, keeping the second layer inactive. After training, both layers are combined into a new model named *alice3*, which is used for inference and classification.

B. ANALYSIS METRICS

During the entire training process, some metrics are calculated by the training module which is later used for model analysis. They are as follows:

1) **Accuracy:** Accuracy is calculated by:

$$\text{Accuracy} = \frac{\text{Number of Correct Predictions}}{\text{Total Number of Predictions}} \tag{12}$$

2) **Mean Reciprocal Rank:** The ability of the model to rank the correct labels among all possible labels. It is calculated by:

$$\text{MRR} = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{\text{rank}_i}, \tag{13}$$

where $rank_i$ is the position of the correct label in the sorted prediction scores for sample i and N is total sample.

3) **Precision and Recall:**
Both metrics are computed per class and then

averaged (e.g., macro-average) as done using `sklearn.metrics`.

Precision (for class c):

$$\text{Precision}_c = \frac{TP_c}{TP_c + FP_c} \tag{14}$$

Recall (for class c):

$$\text{Recall}_c = \frac{TP_c}{TP_c + FN_c} \tag{15}$$

Where:

- TP_c = True Positives for class c
- FP_c = False Positives for class c
- FN_c = False Negatives for class c

In macro-averaging:

$$\text{Precision}_{\text{macro}} = \frac{1}{C} \tag{16}$$

$$\text{Recall}_{\text{macro}} = \frac{1}{C}\sum_{c=1}^{C} \text{Recall}_c \tag{17}$$

4) **Confusion Matrix:** The confusion matrix is a tabular representation:

$$\mathbf{M_{i,j}} \in \mathbb{Z}^{C\times C}$$

Where $M_{i,j}$ = Number of samples with true label i and predicted label j
This provides insight into which classes are being confused with others.

5) **Average Spikes Count:** Computed as:

$$\text{Average Spikes}_l = \frac{1}{B}\sum_{b=1}^{B} S_{l,b} \tag{18}$$

where l is the layer index, B is the total number of batches (or samples), and $S_{l,b}$ is the number of spikes produced by layer l on batch b.

6) **F-1 score:**

$$F_1 = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

C. ANALYSIS OF PROPOSED MODELS RESULTS

The proposed model, referred to as 'SFF-RSTDP' from here on, gave overall positive results when tested on the 2 provided datasets. It is to be mentioned that the data provided below is not of one specific run but average of multiple rounds of training. This applies for all the models mentioned from here on.

1) Initial Results

The table 2 shows the inital performace of our proposed model in all the relevant metrics. The significance of these values have been discussed in details for each respective dataset in section IV-C2 and IV-C3

TABLE 2. SFF-RSTDP Performance Metrics

Dataset	Accuracy	MRR	Precision	Recall	F1 Score
CARER	0.485	0.713	0.468	0.624	0.535
IMDB	0.732	0.866	0.747	0.732	0.739

2) CARER Emotion Dataset

For the CARER emotion dataset, SFF-RSTDP achieved an accuracy of 48.5% and MRR (Mean Reciprocal Rank) of 71.3% while having a precision and recall score of 0.47 and and 0.62 respectively.

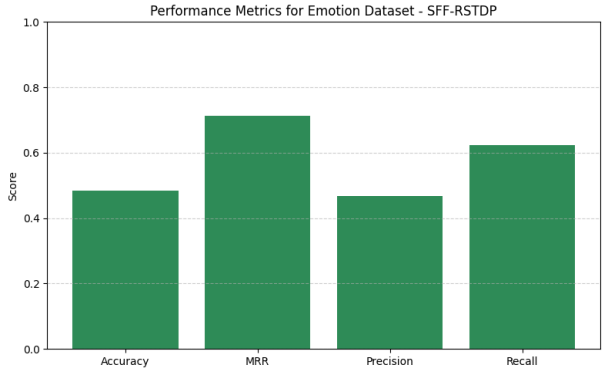

FIGURE 7. Accuracy, MRR, Precision and Recall data for CARER dataset

The figure 7 above states 3 things. Firstly, the model can achieve exceptionally high MRR for an SNN model. Secondly, the proposed model almost achieves an accuracy close to 50% which is a notable achievement when being trained on a multiclass dataset. Lastly, the precision and recall scores are moderate yet consistent with the accuracy achieved by the model. The confusion matrix given below will attest to that fact in figure 8

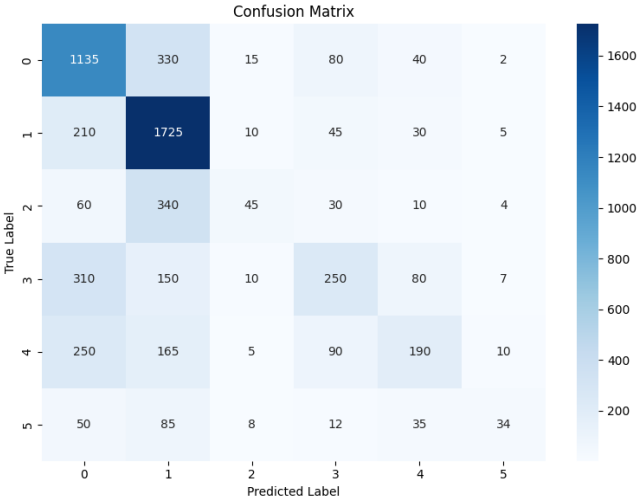

FIGURE 8. SFF-RSTDP Confusion Matrix for CARER Dataset

3) IMDB dataset

For the IMDB dataset, SFF-RSTDP achieved an accuracy of 73.2% and MRR (Mean Reciprocal Rank) of 86.6% while having a precision and recall score of 0.74 and and 0.73 respectively.

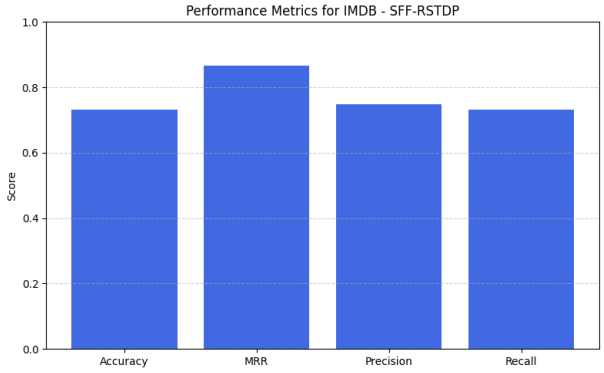

FIGURE 9. Accuracy, MRR, Precision and Recall Data for IMDB dataset

In the case of the IMDB dataset, graphical representation is somewhat similar to CARER dataset. The figure 9 given above indicates the following, that similar to the CARER dataset, the model achieves a very high MRR score with the IMDB dataset. This demonstrates that our model's ability to rank the correct label is exceptionally high for an SNN model. Unlike the CARER dataset, the IMDB dataset is binary-classed. Consequently, it is natural for the model to achieve higher accuracy scores. The figure 9 demonstrates that successfully. Also, unlike the CARER dataset, the precision score of 0.74 is actually slightly higher than its recall score of 0.73. In addition, both the precision and recall score is comparatively high in this case, which results in a more balanced confusion matrix as shown in figure 10

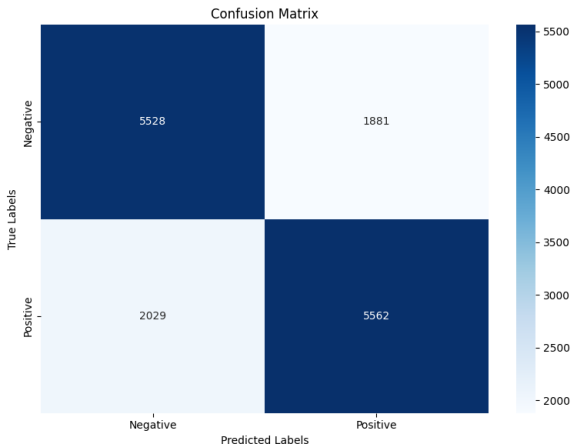

FIGURE 10. SFF-RSTDP Confusion Matrix for IMDB Dataset

D. COMPARISON WITH OTHER MODELS

Given below are the tabulated results for all 5 models and their corresponding graphs. The other SNN models and their benchmark numbers were collected from [44]

TABLE 3. CARER Emotion Dataset Results

Model	Accuracy	MRR	Precision	Recall	F1 Score
ANN	0.567	0.732	0.462	0.354	0.401
SNN	0.340	0.570	0.290	0.350	0.317
SNN-rate	0.380	0.570	0.300	0.510	0.377
SNN-rate-op	0.420	0.608	0.350	0.560	0.429
SFF-RSTDP	0.485	0.713	0.468	0.624	0.535

The table 3 above contains the 4 mentioned metrics for each of the model when the CARER dataset is used. The contents of the table has been analyzed below:

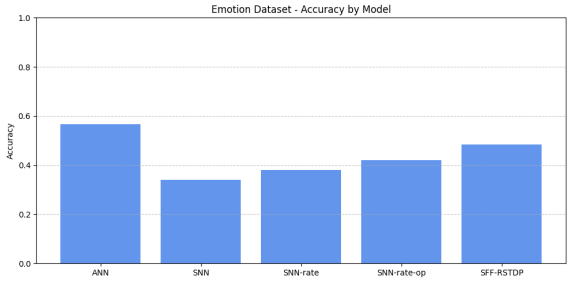

FIGURE 11. Accuracy data for CARER

Figure 11 shows that SFF-RSTDP (48.5% ACC) managed to beat all the other SNN based models [SNN (34% ACC), SNN-rate (38% ACC)] and SNN-rate-op (42% ACC) while failing to compete with the ANN model (56.7% ACC).

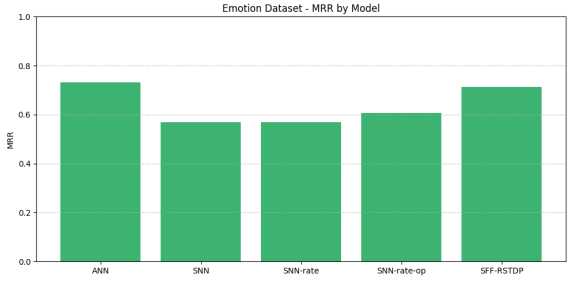

FIGURE 12. MRR for CARER

From the figure 12 above, we can deduce a pattern that while our proposed model SFF-RSTDP (71.3% MRR) consistently beats conventional SNN (57% MRR), SNN-rate (57%) and SNN-rate-op (61.73% MRR), it can't match the results of a conventional ANN model's accuracy or MRR (73.2% MRR) despite getting close to within 2% in this metric of MRR.

The figure 13 demonstrates 2 things. Firstly, it demonstrates SFF-RSTDP's superior ability to be correct when it makes a positive prediction (Precision). Secondly, it shows our models ability to find all the correct positives (Recall). Compared to the values of other models SNN (0.29,0.35), SNN-rate (0.3,0.51), SNN-rate-op (0.35, 0.56) and ANN (0.46, 0.35), only ANN comes close to beating SFF-RSTDP.

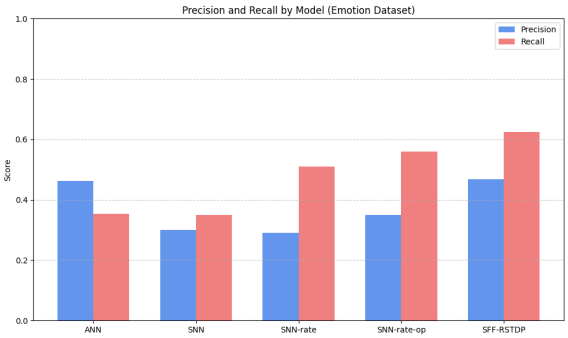

FIGURE 13. Precision and Recall Score for CARER

TABLE 4. Number of Operations for Each Model (CARER Dataset)

Model	Total Operations
ANN	84,736
SNN	1,487
SNN-rate	1,506
SNN-rate-op	1,471
SFF-RSTDP	10,726

At a glance, the table 4 shows that SFF-RSTDP easily beats ANN in terms of reducing energy consumption while getting significantly beaten by the other SNN models. However, the data presented above can be slightly misleading at initial judgement. The SFF-RSTPD and the other SNN models have significant difference in neuron structure and the time steps required to train the model. Due to time constraints and lack of machine resources, the SFF-RSTDP model was run for only one epoch. To compensate for the reduced learning, the time-step applied by the model was 512 compared to 10 of the other models, which significanty bloated the number for average number of operations. However, for a fair comparison, the SNN-rate-op model's time step was increased to 512 for one epoch and the result we obtained was 98,220 which is significatly higher than 10,726. It can be assumed that, if training parameters and conditions were same, the SFF-RSTDP would beat all the SNN based models.

TABLE 5. IMDB Dataset Results

Model	Accuracy	MRR	Precision	Recall	F1 Score
ANN	0.783	0.8910	0.7700	0.8000	0.7840
SNN	0.510	0.7553	0.5000	0.5700	0.5320
SNN-rate	0.490	0.7470	0.4900	1.0000	0.6570
SNN-rate-op	0.648	0.8240	0.5900	0.9100	0.7140
SFF-RSTDP	0.7322	0.8661	0.7473	0.7327	0.7400

The table 5 above contains the 4 mentioned metrics for each of the model when the IMDB dataset is used. The contents of the table has been analyzed below:

The pattern continues in figure 14 as we can see ANN models have the highest accuracy (78.3% ACC) while conventional SNN (51% ACC), SNN-rate (49% ACC) and SNN-rate-op (64.8% ACC) struggles to keep up with SFF-RSTDP (73.2% ACC).

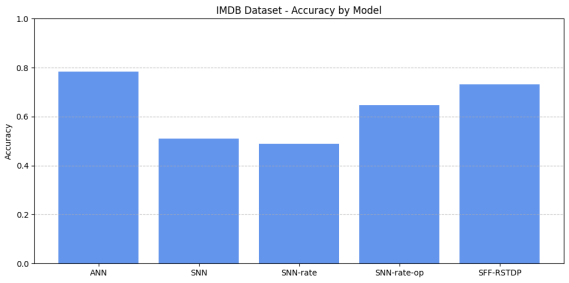

FIGURE 14. Accuracy data for IMDB

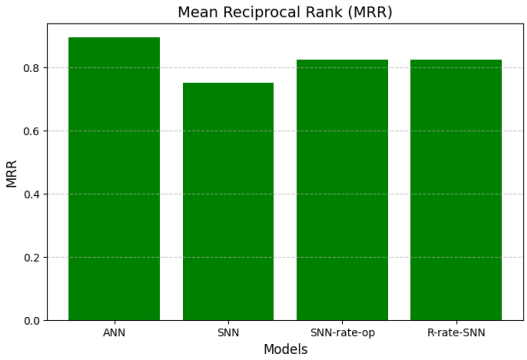

FIGURE 15. MRR for IMDB

Similar to accuracy, figure 15 shows that our model (86.66% MRR) is outperformed by ANN (89.1% MRR). However, it manages to beat the rest of the SNN models very easily.

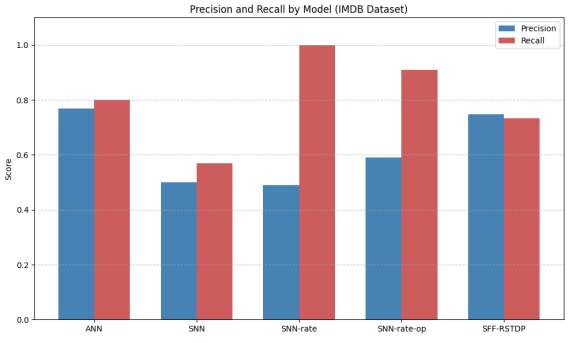

FIGURE 16. Precision and Recall Score for IMDB

Figure 16 shows slightly different results compared to the CARER dataset. In case of the CARER dataset, SFF-RSTDP outperfomred all the other models across the board. In this case however, SFF-RSTDP's precision (0.74) is beat by ANN while its recall score of 0.73 is beat by ANN, SNN-rate and SNN-rate-op.

Despite seemingly not being energy efficient enough to compete with other SNN models, theoretically, its use of

Forward-Forward algorithm will significantly reduce the total number of operations per sample if similar training parameters and conditions were met.

TABLE 6. Number of Operations for Each Model (IMDB Dataset)

Model	Total Operations
ANN	84,244
SNN	1,692
SNN-rate	1,862
SNN-rate-op	1,789
SFF-RSTDP	15,337

After extensive analysis it can be concluded that even though SFF-RSTDP cannot compete with the accuracy and MRR of the ANN model, it significantly reduces the number of operations from 84736 to 10,726 for CARER dataset and from 84224 to 15,337 for IMDB dataset while maintaining respectable scores in the other metrics.

Secondly, compared to non-optimized back propagated SNN and SNN-rate, which are considered to be the most widely used version of the model in current research fields, SFF-RSTDP outperforms them in almost every metric provided. Despite being a deterministic and optimized version of conventional SNN, SNN-rate-op still struggles to compete with the proposed SFF-RSTDP model. The only metric it seemingly falls short in is the Average Number of Operations Per Sample. However, in the analysis section, a detailed explanation has been provided to mitigate this tradeoff. Moreover, if the model was deployed on a neuromorphic hardware, theoretically, the energy consumption reduced by SFF-RSTDP would be much more significant than conventional SNN.

E. ABLATION STUDY

TABLE 7. Ablation Study Results on IMDB

Setting	Acc.	MRR	Prec.	Rec.	F1
Full Model	0.732	0.866	0.747	0.732	0.739
No R-STDP	0.674	0.810	0.692	0.683	0.687
No Surrogate	0.608	0.750	0.631	0.588	0.609
Binary Emb.	0.570	0.710	0.550	0.660	0.600

In order to understand the individual contribution of the components within our biologically inspired SNN model, due to resource limitation, we conducted an ablation study on the IMDB dataset only by disabling some of the key mechanisms like R-STDP, the use of real valued embeddings, and surrogate gradient-based loss. The entire model (SFF-RSTDP) had achieved an accuracy of 73.2%, F1 score of 73.9%, and MRR of 86.6%, while serving as the baseline.

The removal of R-STDP demonstrated a moderated drop in performance with an accuracy of 67.4% and F1 score of 68.7%. This proves the signficance of reward-modulated synaptic updates, as there was a loss in the model's ability to dynamically reinforce the beneficial neuron activations based on the feedback signals. However, the model still retained a certain degree of functionality due the surrogate gradient-based training.

When the surrogate gradient loss was disabled it had a more severe impact, which was an accuracy of 67.4% and a F1 score of 68.7%. This result indicates that while R-STDP supports in guiding the synaptic updates, the surrogate is also very significant in providing consistent gradient-based separation between the positive and negative samples, especially for the deeper layer.

On the other hand, using binary embeddings instead of the real-valued GloVe vectors had extended reduction in the performance with an accuracy of 57.0% and F1 score of 60.0%, which indicates that the reduced semantic resolution of binary encoding leads to a less informative pattern for the spikes. Despite, the model still produced reasonable results, which indicates the robustness of our rate-coding and SNN architecture.

In conclusion, the ablation study validates our design choices, whilst highlighting that every component, R-STDP, real-valued embedding, and surrogate gradient optimization is crucial for achieving a biologically plausible yet a competitive NLP performance as well.

F. DISCUSSION

Following the objectives we set for ourselves in this research, the model we proposed has nearly achieved them all quite successfully. SNNs, especially in the field of NLP, have a long way to go in order to compete with the ANN models. However, the performance of our model effectively shows that it is possible to mitigate that gap in performance with additional use of Reward Modulation and Forward-Forward training paradigms. The table 7 further demonstrates the significance of each part of our proposed pipeline and their contribution to enhancing the performance of SNN. In conclusion, the results of our proposed model lays a solid foundation and baseline for improving SNN model's performance. The final tabulated metric values are shown in table 8.

TABLE 8. Evaluation Metrics and Operation Count for CARER and IMDB Datasets

Dataset	Model	Accuracy	MRR	Precision	Recall	F1 Score	Total Ops
CARER	ANN	0.567	0.732	0.462	0.354	0.401	84,736
	SNN	0.340	0.570	0.290	0.350	0.317	1,487
	SNN-rate	0.380	0.570	0.300	0.510	0.377	1,506
	SNN-rate-op	0.420	0.608	0.350	0.560	0.429	1,471
	SFF-RSTDP	0.485	0.713	0.468	0.624	0.535	10,726
IMDB	ANN	0.783	0.891	0.7700	0.8000	0.784	84,244
	SNN	0.510	0.7553	0.5000	0.5700	0.532	1,692
	SNN-rate	0.490	0.7470	0.4900	1.0000	0.657	1,862
	SNN-rate-op	0.648	0.8240	0.5900	0.9100	0.714	1,789
	SFF-RSTDP	0.7322	0.8661	0.7473	0.7327	0.740	15,337

V. CONCLUSION

Our research presents the R-STDP SFF model, a biologically inspired, decentralized learning framework that combines Reward-Modulated Spike-Timing Dependent Plasticity (R-STDP) with the Forward-Forward (FF) algorithm for natural language classification tasks. By applying this model to sentiment analysis and affective computing datasets like IMDB and CARER, the study demonstrates a promising intersection of biological plausibility, computational efficiency, and layer-wise local learning without reliance on global backpropagation. The FF algorithm enables each layer to evaluate its output independently using a spike-based "goodness" metric, mimicking cortical-level autonomy in the brain. Coupled with R-STDP's dopamine-modulated reward system, the model learns adaptively through time-sensitive weight updates that reflect synaptic plasticity in biological neurons. The architecture also improves energy efficiency by promoting sparse, binary spiking activity, making it potentially well-suited for future deployment on neuromorphic hardware.

However, the approach comes with several tradeoffs and limitations. The most prominent is the performance gap compared to state-of-the-art artificial neural networks, particularly Transformer-based models. These outperform SNNs in accuracy but lack energy efficienct training paradigms. The scope of this study was also limited to specific datasets, which restricts broader generalization. Additionally, due to lack of access to neuromorphic platforms like Intel Loihi or IBM TrueNorth, hardware-level validation of spike efficiency and latency remains unexplored. Lastly, the model structure is still in a foundational stage, and there is room for architectural improvements.

Looking forward, future research should focus on expanding the dataset diversity to include more complex, multilingual, or multi-domain text sources. There is also strong potential in hybrid models that integrate SNN layers with Transformer components, aiming to combine the benefits of biological plausibility and state-of-the-art accuracy. Finally, deploying the model on actual neuromorphic hardware would provide a more accurate assessment of its energy and latency advantages, moving closer to practical, brain-like cognitive systems.

REFERENCES

[1] T. Iakymchuk, A. Rosado-Muñoz, J. F. Guerrero-Martínez, M. Bataller-Mompeán, and J. V. Francés-Víllora, "Simplified spiking neural network architecture and STDP learning algorithm applied to image classification," EURASIP J. Image Video Process., vol. 2015, pp. 1–11, Dec. 2015.

[2] J. D. Nunes, M. Carvalho, D. Carneiro, and J. S. Cardoso, "Spiking neural networks: A survey," IEEE Access, vol. 10, pp. 60738–60764, Jun. 2022.

[3] E. O. Neftci, H. Mostafa, and F. Zenke, "Surrogate gradient learning in spiking neural networks: Bringing the power of gradient-based optimization to spiking neural networks," IEEE Signal Process. Mag., vol. 36, no. 6, pp. 51–63, Nov. 2019.

[4] S. R. Kheradpisheh, M. Ganjtabesh, S. J. Thorpe, and T. Masquelier, "STDP-based spiking deep convolutional neural networks for object recognition," Neural Netw., vol. 99, pp. 56–67, Mar. 2018.

[5] Q. Zhou, C. Ren, and S. Qi, "An imbalanced R-STDP learning rule in spiking neural networks for medical image classification," IEEE Access, vol. 8, pp. 224162–224177, Dec. 2020.

[6] M. Mozafari, S. R. Kheradpisheh, T. Masquelier, A. Nowzari-Dalini, and M. Ganjtabesh, "First-spike-based visual categorization using reward-modulated STDP," IEEE Trans. Neural Netw. Learn. Syst., vol. 29, no. 12, pp. 6178–6190, Dec. 2018.

[7] M. Mozafari, M. Ganjtabesh, A. Nowzari-Dalini, S. J. Thorpe, and T. Masquelier, "Bio-inspired digit recognition using reward-modulated spike-timing-dependent plasticity in deep convolutional networks," Pattern Recognit., vol. 94, pp. 87–95, Oct. 2019.

[8] R. A. Knipper, K. Mishty, M. Sadi, and S. K. Karmaker, "SNNLP: Energy-efficient natural language processing using spiking neural networks," arXiv preprint, arXiv:2401.17911, Jan. 2024.

[9] Y. Zeng, D. Zhao, F. Zhao, G. Shen, Y. Dong, E. Lu, Q. Zhang, Y. Sun, Q. Liang, Y. Zhao, Z. Zhao, H. Fang, Y. Wang, Y. Li, X. Liu, C. Du, Q. Kong, Z. Ruan, and W. Bi, "BrainCog: A spiking neural network based, brain-inspired cognitive intelligence engine for brain-inspired AI and brain simulation," Patterns (N Y), vol. 4, p. 100789, Jul. 2023.

[10] Z. Shao, X. Fang, Y. Li, C. Feng, J. Shen, and Q. Xu, "EICIL: Joint excitatory inhibitory cycle iteration learning for deep spiking neural networks," Adv. Neural Inf. Process. Syst., vol. 36, pp. 32117–32128, Dec. 2023.

[11] Y. Chen, Z. Yu, W. Fang, Z. Ma, T. Huang, and Y. Tian, "State transition of dendritic spines improves learning of sparse spiking neural networks," in Proc. Int. Conf. Mach. Learn., pp. 3701–3715, Jul. 2022.

[12] J. Ding, B. Dong, F. Heide, Y. Ding, Y. Zhou, B. Yin, and X. Yang, "Biologically inspired dynamic thresholds for spiking neural networks," Adv. Neural Inf. Process. Syst., vol. 35, pp. 6090–6103, Dec. 2022.

[13] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," Adv. Neural Inf. Process. Syst., vol. 26, Dec. 2013.

[14] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching word vectors with subword information," Trans. Assoc. Comput. Linguist., vol. 5, pp. 135–146, Dec. 2017.

[15] T. Mikolov, "Efficient estimation of word representations in vector space," arXiv preprint, arXiv:1301.3781, Jan. 2013.

[16] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in Proc. 2014 Conf. Empir. Methods Nat. Lang. Process. (EMNLP), pp. 1532–1543, Oct. 2014.

[17] A. Conneau, D. Kiela, H. Schwenk, L. Barrault, and A. Bordes, "Supervised learning of universal sentence representations from natural language inference data," arXiv preprint, arXiv:1705.02364, May 2017.

[18] D. Cer, "Universal sentence encoder," arXiv preprint, arXiv:1803.11175, Mar. 2018.

[19] J. Tissier, C. Gravier, and A. Habrard, "Near-lossless binarization of word embeddings," in Proc. AAAI Conf. Artif. Intell., vol. 33, pp. 7104–7111, Jul. 2019.
[20] A. Vaswani, "Attention is all you need," Adv. Neural Inf. Process. Syst., Dec. 2017.
[21] T. Iakymchuk, A. Rosado-Muñoz, J. F. Guerrero-Martínez, M. Bataller-Mompeán, and J. V. Francés-Víllora, "Simplified spiking neural network architecture and STDP learning algorithm applied to image classification," EURASIP J. Image Video Process., vol. 2015, pp. 1–11, Dec. 2015.
[22] Y. Dong, D. Zhao, Y. Li, and Y. Zeng, "An unsupervised STDP-based spiking neural network inspired by biologically plausible learning rules and connections," Neural Netw., vol. 165, pp. 799–808, Aug. 2023.
[23] Y. Hao, X. Huang, M. Dong, and B. Xu, "A biologically plausible supervised learning method for spiking neural networks using the symmetric STDP rule," Neural Netw., vol. 121, pp. 387–395, Jan. 2020.
[24] J. C. V. Tieck, P. Becker, J. Kaiser, I. Peric, M. Akl, D. Reichard, A. Roennau, and R. Dillmann, "Learning target reaching motions with a robotic arm using brain-inspired dopamine modulated STDP," in Proc. 2019 IEEE 18th Int. Conf. Cogn. Informat. Cogn. Comput. (ICCI*CC), pp. 54–61, Jul. 2019.
[25] Z. Bing, C. Meschede, K. Huang, G. Chen, F. Rohrbein, M. Akl, and A. Knoll, "End to end learning of spiking neural network based on R-STDP for a lane keeping vehicle," in Proc. 2018 IEEE Int. Conf. Robot. Autom. (ICRA), pp. 4725–4732, May 2018.
[26] M. Białas, M. M. Mirończuk, and J. Mańdziuk, "Biologically plausible learning of text representation with spiking neural networks," in Proc. Int. Conf. Parallel Problem Solving Nat., pp. 433–447, Sep. 2020.
[27] Z. Bing, C. Meschede, K. Huang, G. Chen, F. Rohrbein, M. Akl, and A. Knoll, "End to end learning of spiking neural network based on R-STDP for a lane keeping vehicle," in Proc. 2018 IEEE Int. Conf. Robot. Autom., May 2018.
[28] L. N. Andersen and T. S. Haus, "Dopamine modulated STDP and reinforcement learning in an embodied context," in Proc., 2013.
[29] M. Yao, H. Gao, G. Zhao, D. Wang, Y. Lin, Z. Yang, and G. Li, "Temporal-wise attention spiking neural networks for event streams classification," arXiv preprint, arXiv:2107.11711, Jul. 2021.
[30] M. Pfeiffer and T. Pfeil, "Deep learning with spiking neural networks: Opportunities and challenges," Front. Neurosci., vol. 12, p. 774, Oct. 2018.
[31] O. Y. Sinyavskiy, "Training of spiking neural networks based on information theoretic costs," arXiv preprint, arXiv:1602.04742, Feb. 2016.
[32] M. Mozafari, S. R. Kheradpisheh, T. Masquelier, A. Nowzari-Dalini, and M. Ganjtabesh, "First-spike-based visual categorization using reward-modulated STDP," IEEE Trans. Neural Netw. Learn. Syst., vol. 29, no. 12, pp. 6178–6190, Dec. 2018.
[33] A. Maas, R. Daly, P. Pham, D. Huang, A. Ng, and C. Potts, "Learning word vectors for sentiment analysis," in Proc., pp. 142–150, Jan. 2011.
[34] E. Saravia, H.-C. Liu, Y.-H. Huang, J. Wu, and Y.-S. Chen, "CARER: Contextualized affect representations for emotion recognition," in Proc., pp. 3687–3697, Jan. 2018.
[35] M. Ghader, S. R. Kheradpisheh, B. Farahani, and M. Fazlali, "Backpropagation-free spiking neural networks with the forward-forward algorithm," arXiv preprint, arXiv:2502.20411, Feb. 2025.
[36] E. Lemaire, L. Cordone, A. Castagnetti, P.-E. Novac, J. Courtois, and B. Miramond, "An analytical estimation of spiking neural networks energy efficiency," Neural Inf. Process., pp. 574–587, Apr. 2023.
[37] G. Hinton, "The forward-forward algorithm: Some preliminary investigations," arXiv preprint, arXiv:2212.13345, Dec. 2022.
[38] E. Strubell, A. Ganesh, and A. McCallum, "Energy and policy considerations for deep learning in NLP," arXiv preprint, arXiv:1906.02243, Jun. 2019.
[39] Y. Wu, X. Hu, L. Liang, Y. Ding, G. Li, G. Zhao, P. Li, and Y. Xie, "Rethinking the performance comparison between SNNs and ANNs," Neural Netw., vol. 121, Sep. 2019.
[40] E. O. Neftci, H. Mostafa, and F. Zenke, "Surrogate gradient learning in spiking neural networks: Bringing the power of gradient-based optimization to spiking neural networks," IEEE Signal Process. Mag., vol. 36, no. 6, pp. 51–63, Nov. 2019.
[41] S. Gandhi, R. Gala, J. Kornberg, and A. Sridhar, "Extending the forward forward algorithm," arXiv preprint, arXiv:2307.04205, Jul. 2023.
[42] R.-J. Zhu, Q. Zhao, G. Li, and J. K. Eshraghian, "SpikeGPT: Generative pre-trained language model with spiking neural networks," arXiv preprint, arXiv:2302.13939, Jun. 2024.
[43] J. Vitay, "IC_IRP_2023_SpikingForwardForward," GitHub repository, 2023.
[44] A. Knipper, "SNNLP: Spiking neural networks for natural language processing," GitHub repository, 2023.
[45] L.-Y. Niu, Y. Wei, W.-B. Liu, J.-Y. Long, and T.-H. Xue, "Research progress of spiking neural network in image classification: A review," Appl. Intell., vol. 53, pp. 1–25, Mar. 2023.
[46] S. K. Esser, R. Appuswamy, P. Merolla, J. V. Arthur, and D. S. Modha, "Backpropagation for energy-efficient neuromorphic computing," Adv. Neural Inf. Process. Syst., vol. 28, Dec. 2015.