Motivation

Under the extensive, ever-growing umbrella of artificial intelligence rests dozens of interdisciplinary research topics, brand-new subfields, and cutting-edge applications that benefit virtually every industry. Machine learning has always been a very exciting and promising field to work in; its accomplishments are impressive even before deep neural networks when it wasn't much more than statistical approximation. Several years ago I read an article about artificial intelligence in the Wall Street Journal for the first time, whereas now I don't usually go more than a day without stumbling on something. While media conglomerates are notorious for preying on the trepidation of the average American for the sake of views, I see the prevalence of AI in the newspaper as a well-deserved testament to the industry's unprecedented success. ML methods and algorithms will continue to improve towards pure, unadulterated artificial intelligence as a consequence of our constantly developing understanding of the brain's inherent subsystems. Unfortunately true biologically plausible machines aren't much more than a fairytale as our innate neurophysiological processes are just too complicated to accurately model with Von-Neumann architectures. Our unique ability to understand language is hot off the press relative to other evolutionary adaptations[1], language comprehension can be looked at as one of the most challenging branches of artificial intelligence. This is backed up by the fact that language comprehension requires simultaneous operation of the different types of language (such as knowledge about letters, spelling, grammar, word meaning)[2]. The organic complexity of natural language processing potentially births the need for a neuronal composition a bit further on up the road towards biological realism than the artificial neural network. In this project, we evolve our perception of language in the brain by investigating recent works at the intersection of next-generation neural frameworks and language comprehension models.

Problem Formulation

ChatGPT's unique ability to generate accurate, human-like responses to input questions is why it is the fastest growing app of all time, reaching 100 million users in two months. The chatbot is forecasted to bring in roughly 250 million in revenue by year's end and over a billion by 2024[3]. The AI chatbot was developed by OpenAI and built on a family of large language models (LLMs) collectively known as GPT-3. It is a language processing model trained on a huge text dataset scraped from the internet, totaling 570GB and 300 billion words[4], that is able to understand and respond to natural language prompts and questions. Augmenting any arbitrary dataset to improve model performance is accomplished by increasing the size and diversity of its training dataset[5]. A jump in data complexity typically results in a decrease of model efficiency, which creates concerns about the computational efficiency of the language model. Thus the eclectic mix of writing styles found in the dataset allows the model to better understand and mimic human language while also increasing its complexity. Although the computational efficiency of ANN-based LLMs isn't a huge concern today, it will be tomorrow. Therefore we focus on papers which see an LLM exploit the advantages of spiking neural networks, namely SpikeGPT.

Explanation of Paper - The following section, Spiking Up, acts as an introduction to spiking neural networks. Next we have the SpikeGPT section, which goes through and explains each step up the model's process, under the assumption the reader is familiar with transformers and self-attention. The last section titled, Discussion, comprises two subsections: Limitations, which quickly lists the major drawbacks of SNN-based models in general, and Looking Ahead, in which I speculate on the future of a few of the research subfields introduced.

Spiking Up

The Road To Biologically Plausible Machines

While there are a million different ways in which machine learning can be used to benefit society, the underlying goal of the field is to put the human brain on a computer. This subfield, dubbed computational neuroscience, works to understand the principles that govern the development, structure, and cognitive abilities of the nervous system[6]. Thus computational neuroscience can be thought of as a slow march on the yellow brick road of biological realism, starting at the Von Neumann architecture with a destination of true artificial intelligence. This march can be better explained by shedding light on how the neural network has evolved over time since its invention. As seen in Table 1 and the subsequent summaries, the history of the NN can be partitioned into three generations, each of which is explored. But since contemporary models and algorithms in computational neuroscience are obviously derived from the latest generation of biological machines, we introduce its core concepts in the next section.

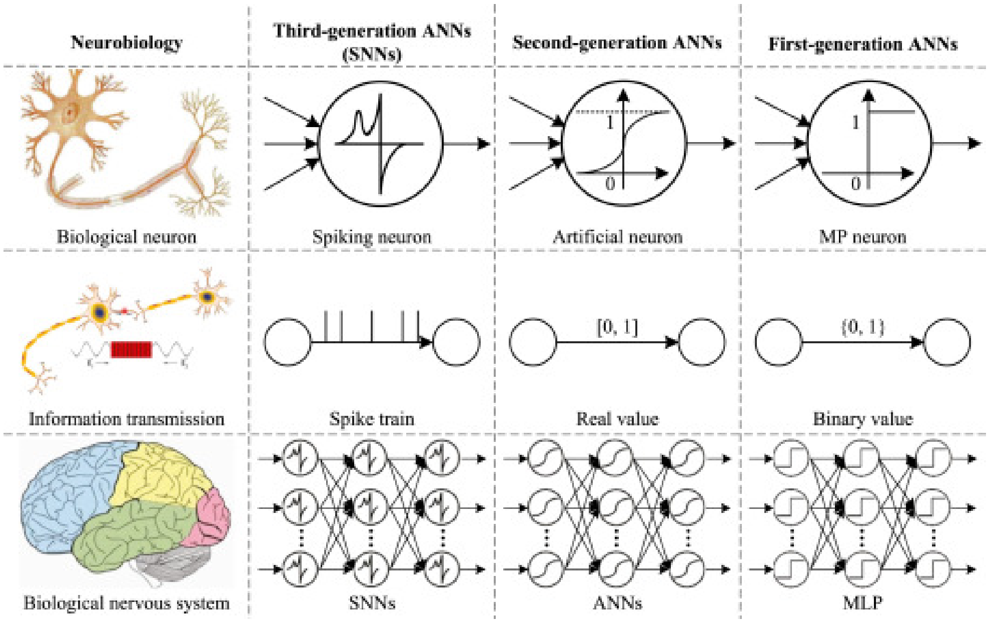


Figure 4: History of Neural Networks

1. **First-Generation** - This network was based on neurons that acted as threshold gates or perceptrons. Its output was binary and therefore did not involve a nonlinear activation function.
2. **Second-Generation** - your standard feedforward artificial neural network (ANN) along with its dozens of flavors (RNN, CNN, etc).
3. **Third-Generation** - employs spiking neurons, a.k.a integrate-and-fire neurons, which more closely model the biological neurons' activity relative to the first two generations.

Spikes in the Brain[7]

What? A spike, or action potential, is a short burst of electrical activity which occurs when a neuron sends information down an axon, away from the cell body. A spike, roughly 100 mV in amplitude as seen in Figure 5, can be thought of as the electric current being passed to downstream neurons. A temporal sequence of these action potentials generated by the neuron are called spike trains. Neurons send messages electrochemically, or via electrically charged chemicals called ions. The most important ions in the nervous system are sodium, potassium, calcium, and chloride.

Why? Our sensory neurons encode information about our surrounding environment, like our five senses, into sequences of voltage spikes with distinct temporal patterns called spike trains. The brain communicates in the universal language of the *spike*. This form of information representation and communication allows biological neurons to disseminate information extremely efficiently, its main advantage over second generation networks. Therefore understanding the lower-level characteristics of these neurons is critical to understanding not only how our brain ingests information, but also how it is stored and used.

How? Neurons are surrounded by a thin lipid bilayer which acts as an insulator for its internal conductive saline solution from the extracellular medium. This membrane is embedded with numerous types of ion channels which allows it to control the flow of ions in and out of the neuron. These voltage-gated ion channels rapidly depolarize and repolarize the transmembrane potential by manipulating the throughput of the four previously mentioned ions. This is how spikes are generated in the brain. The following figure provides more insight into the underlying stages of an action potential:

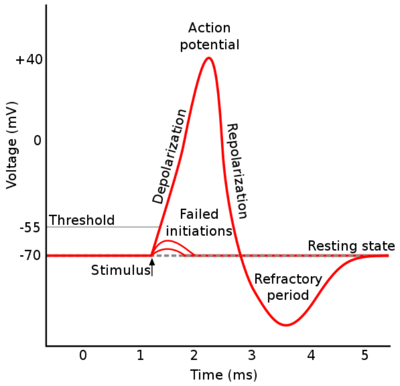
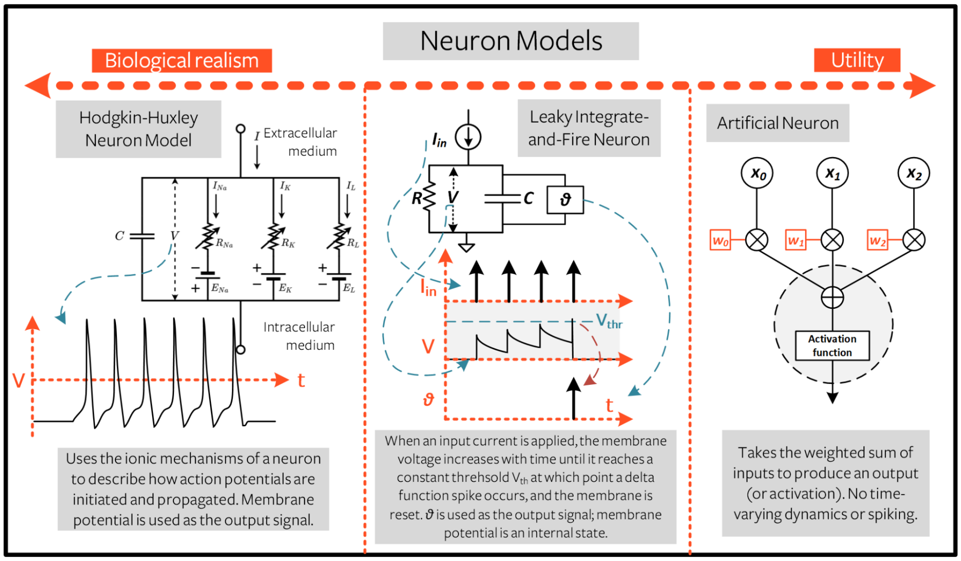


Figure 5: Labeled Action Potential Graph

Neuron Models

To properly compare and contrast the extensive and eclectic mix of neuron models in existence, we map them to the biological-realism domain. In other words, we correlate the neuron models based on a single, unifying tradeoff metric: plausibility vs utility. The tradeoff defines the inverse relationship between the neuron models' bio-realism and their actual utility to deep learning on Von Neumann machines. The bullet points below introduce each of the neuron models presented in the following figure:



Uses the ionic mechanisms of a neuron to describe how action potentials are initiated and propagated. Membrane potential is used as the output signal.

When an input current is applied, the membrane voltage increases with time until it reaches a constant threhsold V_{th} at which point a delta function spike occurs, and the membrane is reset. ϑ is used as the output signal; membrane potential is an internal state.

Takes the weighted sum of inputs to produce an output (or activation). No time-varying dynamics or spiking.

Figure 6: Neuron Model Trade off := Bio-realism <-> Utility

Artificial Neuron[8] - The McCulloch-Pitts neuron was conceived by Warren McCulloch and Walter Pitts in their 1943 paper, "A Logical Calculus of Ideas Immanent in Nervous Activity", and is considered the building block of the first generation NN. This model argues that neurons with a binary threshold activation function are analogous to first order logic tables (ie AND & OR).The artificial neuron model, derived from the McCulloch-Pitts neuron, is the building block of the second generation neural networks defined earlier in this section. Also referred to as the perceptron, this model is considered the foundation of deep learning as it incorporates higher-order logic (i.e. XOR or the multi-layer network).

Hodgkin-Huxley Model[9] - Regarded as one of the great achievements of 20th century biophysics, the Hodgkin-Huxley model describes how the action potentials in neurons are initiated and propagated, represented by a set of nonlinear differential equations. This model gives insight into the relationship between the flow of ionic currents across the neuronal cell membrane and the membrane voltage of the cell. Since this mathematical model has a high degree of biophysical accuracy, it is very complex and therefore not particularly useful to deep learning on Von Neumann machines.

Leaky Integrate-And-Fire Neuron Model[10] - We are looking for a good balance between these two extremes, enter the LIF neuron model. The most important distinction of the LIF neuron is that it extends its artificial counterpart to the temporal domain. Which just means that information is encoded within the timing of spikes, rather than in the individual spike. Instead of passing the sum of weighted inputs directly to the activation function, as with the artificial neuron, the LIF model integrates the input over time (with a leak). If the integration exceeds some threshold value then the LIF neuron spikes, thus that threshold is very important to the pattern of action potentials. The different components of the LIF neuron can be modeled as a resistor-capacitor (RC) circuit as seen in Figure 7a. The membrane potential equation can be found using Kirchhoff's current law, which states that the sum of all currents flowing into a node is equal to the sum of all currents flowing out of the node. The derivation for this equation can be seen in detail in Figure 7b below:

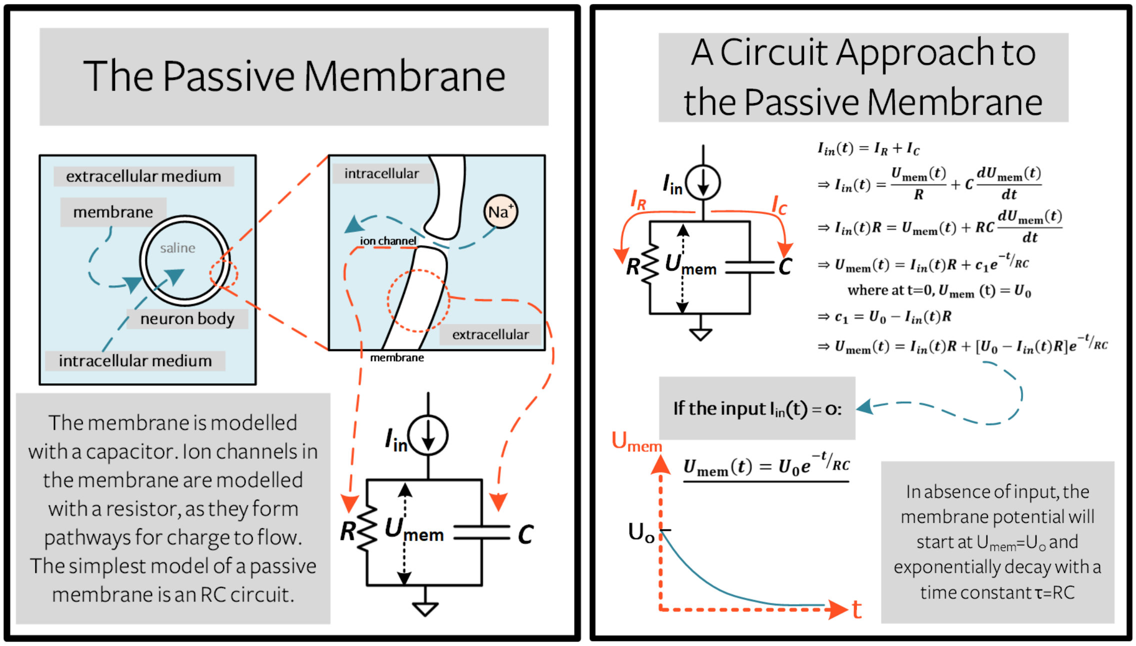

The Passive Membrane
extracellular medium
membrane
saline
neuron body
intracellular medium
intracellular
Na+
ion channel
extracellular
membrane
The membrane is modelled with a capacitor. Ion channels in the membrane are modelled with a resistor, as they form pathways for charge to flow. The simplest model of a passive membrane is an RC circuit.
I_{in}
R
U_{mem}
C
A Circuit Approach to the Passive Membrane
I_{in}
I_R
I_C
R
U_{mem}
C
$I_{in}(t) = I_R + I_C$
$\Rightarrow I_{in}(t) = \frac{U_{\text{mem}}(t)}{R} + C\frac{dU_{\text{mem}}(t)}{dt}$
$\Rightarrow I_{in}(t)R = U_{\text{mem}}(t) + RC\frac{dU_{\text{mem}}(t)}{dt}$
$\Rightarrow U_{\text{mem}}(t) = I_{in}(t)R + c_1 e^{-t/RC}$
where at t=0, U_{mem} (t) = U_0
$\Rightarrow c_1 = U_0 - I_{in}(t)R$
$\Rightarrow U_{\text{mem}}(t) = I_{in}(t)R + [U_0 - I_{in}(t)R]e^{-t/RC}$
If the input I_{in}(t) = 0:
U_{mem}
$U_{\text{mem}}(t) = U_0 e^{-t/RC}$
U_0
t
In absence of input, the membrane potential will start at U_{mem}=U_0 and exponentially decay with a time constant τ=RC

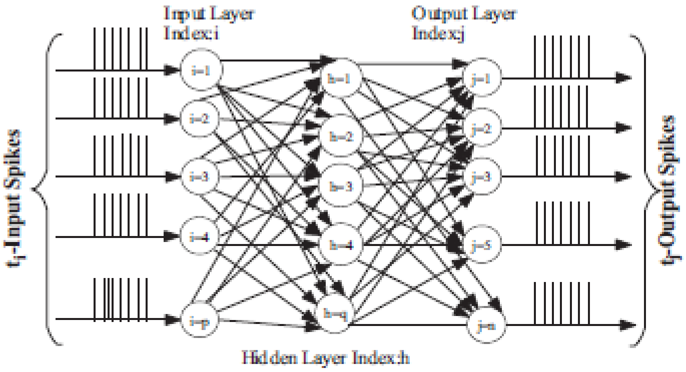
Figure 7: The Passive Membrane - (a) mapping the passive membrane to an electric circuit, and (b) applying circuit analysis to find the voltage membrane equation

This is not meant to be a comprehensive examination of the LIF neuron model, primarily because there is too much information to go over within the given time constraints. I highly advise doing your own research on the concepts I present if you want an exhaustive understanding.

The [Feedforward] Spiking Neural Network

Architectures[11]

The generic feedforward SNN architecture, as illustrated in the next figure, is pretty similar to that of the ANN in terms of layers: both have input, hidden, and output layers. The only difference worth mentioning in this section is that both the input and output of this network are in spike form.



Architecture of a multilayer spiking neural network.

Figure 8: FF-SSN

But neurons in the brain are innately recurrent, which means they have a lot of feedback connections, so we can reformulate the spiking neuron into a discrete, recursive form. This recurrent form is well set up to exploit the developments in training RNNs and sequence based models, as seen in the following figure:

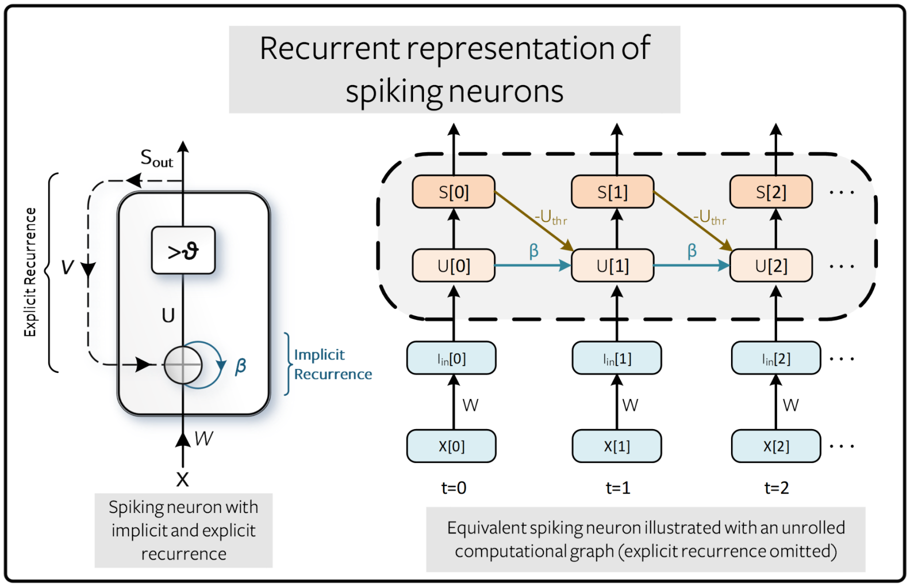

Recurrent representation of spiking neurons
Sout
Explicit Recurrence
V
>ϑ
U
β
Implicit Recurrence
W
X
Spiking neuron with implicit and explicit recurrence
S[0]
S[1]
S[2]
-Uthr
-Uthr
U[0]
U[1]
U[2]
β
β
Iin[0]
Iin[1]
Iin[2]
W
W
W
X[0]
X[1]
X[2]
t=0
t=1
t=2
Equivalent spiking neuron illustrated with an unrolled computational graph (explicit recurrence omitted)

Figure 9: R-SNN

Training[12]

SNNs are difficult to train with backpropagation due to something called the dead neuron problem, which refers to the inherent non-differentiability of spikes. As of now there is no learning algorithm built expressly for SNNs, which is an open problem in computational neuroscience. Although this is probably one of the bigger drawbacks of SNNs, once an SNN learning algorithm is designed, the ANN will be considered outdated. In the meantime there are a few short-term workarounds which allow SNN training, we will focus on the most popular one: surrogate gradient descent. This technique applies some smoothing function (ie sigmoid, fast sigmoid, etc) to the Heaviside step function to ensure its gradient isn't almost always zero. This backalley technique to SNN training is explained visually in the coming figure:

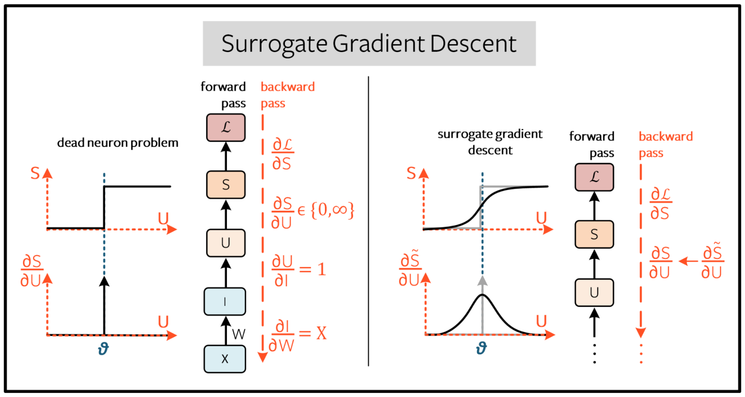

Surrogate Gradient Descent
dead neuron problem
S
U
$\frac{\partial S}{\partial U}$
U
ϑ
forward pass
backward pass
$\mathcal{L}$
S
U
I
W
X
$\frac{\partial \mathcal{L}}{\partial S}$
$\frac{\partial S}{\partial U} \in \{0,\infty\}$
$\frac{\partial U}{\partial I} = 1$
$\frac{\partial I}{\partial W} = X$
surrogate gradient descent
S
U
$\frac{\partial \tilde{S}}{\partial U}$
U
ϑ
forward pass
backward pass
$\mathcal{L}$
S
U
$\frac{\partial \mathcal{L}}{\partial S}$
$\frac{\partial S}{\partial U} \leftarrow \frac{\partial \tilde{S}}{\partial U}$

Figure 10: Surrogate Gradient Descent

The figure above only depicts the gradient calculation for a single timestep, but the backpropagation through time (BPTT) algorithm computes the gradient of the loss with respect to all descendants and sums them. The BPTT algorithm is used to train ANNs and therefore SNNs because there's no existing spiking learning algorithm.

SNNs Versus ANNs

The advantage of the biological neuron relative to its artificial sister lies in its ability to disseminate information extremely efficiently. SNNs employ event-driven processing which means only new information is transmitted, as opposed to transmitting at every propagation cycle. Event-driven processing contributes to the firing sparsity and power efficiency of spiking neural networks, making them more potent than their artificial counterparts[13].

SpikeGPT[14]

Paper Contributions

- Largest SNN trained to date in terms of parameter count, with the largest version at 260 parameters (4x more than the previous holder of that title)
- The first demonstration SNNs can be used for generative tasks
- An integrated, SNN-compatible recurrence similar to a transformer, but doesn't have the same complexity (self-attention has quadratic complexity)
- A demonstration that SNNs can achieve results comparable to the transformer, but with a more energy-efficient approach

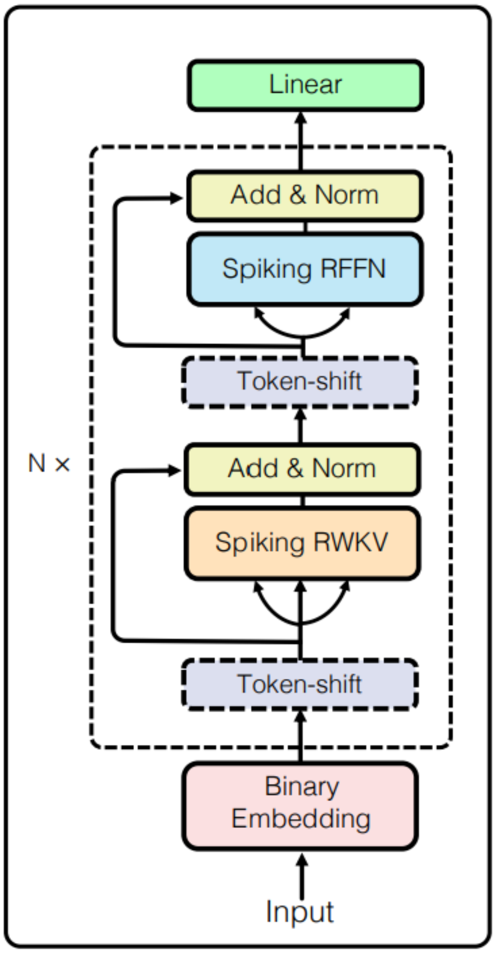


Figure 11: SpikeGpt Architecture

How Does it Work?

- **Binary Embedding** - maps the continuous outputs from the embedding layer from a continuous space to a binary space, as spike form is modeled using binary variables.
- **Token-shift** - this operator combines information from the original token with info from the global context, this ensures more contextual information per token
- **Add & Norm** - sums and normalizes the outputs of spiking modules

- **Spiking RWKV**

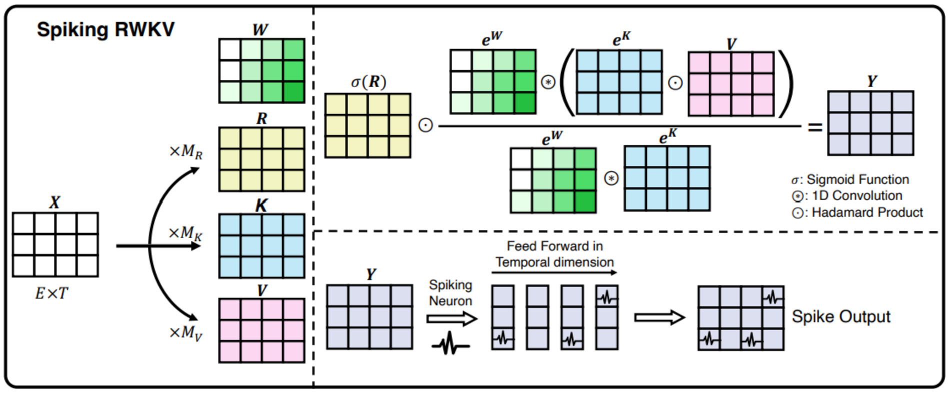


Figure 12: Spiking RWKV Steps in Matrix Form

Receptance Weighted Key Value (RWKV) - Inspired by the attention-free transformer, RWKV acts as a replacement for self-attention. This clever language model exploits the advantages of both transformer training (i.e. in parallel) and RNN inference[15]. Intuitively, RWKV represents a function which takes a token and a state, and outputs a probability distribution over the next token, and a new state. The key and value vectors are the same as in QKV attention but now the query vector is replaced with the weighted receptance vector, which indicates the acceptance of past information. Therefore receptance can also be thought of as a forget gate that eliminates unnecessary historical information.

Vanilla RWKV - Given an input token-shifted embedding vector, similar to self-attention, RWKV first applies a linear transformation to generate the time-varying R, K, V matrices. This first step is illustrated in the leftmost section in Figure 12 above. The second step, as drawn in the top section of Figure 12, involves applying the following operation:

$$Y_t = \sigma(R_t) \odot \frac{\sum_{i=1}^{t} \exp(W_{(T-i+1)}) \odot \exp(K_i) \odot V_i}{\sum_{i=1}^{t} \exp(W_{(T-i+1)}) \odot \exp(K_i)}$$

Equation 1: Vanilla RWKV; where sigma is the nonlinear softmax function

RWKV Enabled SNN - The serial RNN formulation of RWKV is expressed below:

$$Y[t+1] = \sigma(RX[t]) \cdot \frac{\exp(KY[\mathrm{t}]) \cdot (VY[\mathrm{t}]) + \exp(W) \cdot A[\mathrm{t}]}{\exp(KY[\mathrm{t}]) + \exp(W) \cdot B[\mathrm{t}]}$$

$$A[t] = \exp(KY[\mathrm{t}-1]) \cdot (VY[\mathrm{t}-1]) + \exp(W) \cdot A[\mathrm{t}-1]$$

$$B[t] = \exp(KY[\mathrm{t}-1]) + \exp(W) \cdot B[\mathrm{t}-1]$$

Equation 2: Serial RNN RWKV; where t denotes the timestep, and A & B are hidden states

The output, Y, is passed as input to the LIF neuron model as seen in the bottom section of Figure 12 above and exemplified below in Equation 3:

$$\begin{cases} U[t] = H[t] + \beta(Y[t] - (H[t-1] - U_{\text{reset}})) \\ S[t] = \Theta(U[t] - U_{\text{threshold}}) \\ H[t] = U[t] \cdot (1 - S[t]) \end{cases}$$

Equation 3: LIF Neuron Model; where U is the membrane potential, H denotes the spike emission reset process, beta is the decay scalar, S is the binary spiking tensor, and theta denotes the Heaviside function.

- **Spiking Receptance Feedforward Network**

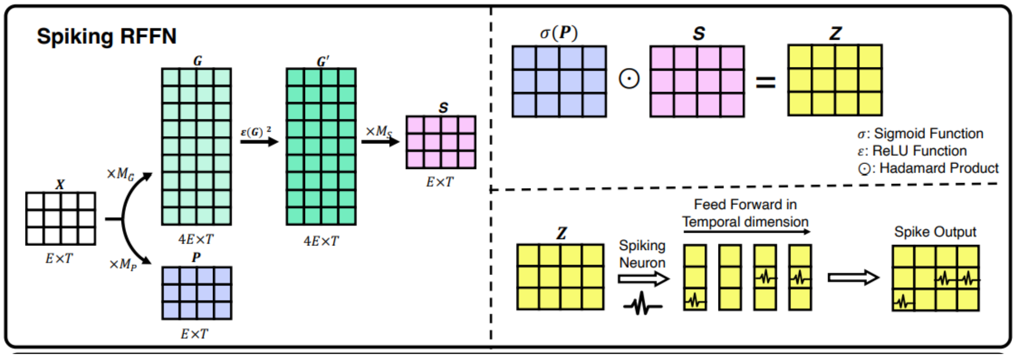


Figure 13: Spiking RFFN Steps in Matrix Form

Each block in our model contains a classic feed-forward network with a gating mechanism, which is applied to normalized, token-shifted output of each spiking RWKV module. This SRFFN module consists of three linear transformations with squared *ReLU* activations as follows:

$$Y'[t] = \sigma(M_P X[t]) \odot M_S(ReLU^2(M_G X[t]))$$

Equation 4: SRFFN Equation; where Y' denotes the input to the LIF neuron model and the M_* matrices are learnable parameters of the linear transformations. The softmax function dictates how much information the model processes, similar to the gated linear unit (GLU).

Discussion

Limitations

- **Learning Algorithms**[16] - There are a lack of training algorithms that exploit the capabilities of spiking neurons, e.g. efficient time codes, as they are more difficult to design and analyze. This is due to the asynchronous and discontinuous way of computing which makes direct application of successful back propagation techniques, as used for ANNs, difficult.
- **Spike Representation**[16] - Even though we are sure that the brain uses the spike as means of communication between neurons, we have no idea how the brain encodes information into spike form. Current approaches to spike encoding aren't bad, but it would be ignorant to assume that they measure up to the brain's encoding scheme in any way.
- **Biological Plausibility Computational Utility Tradeoff**[16] - In the short term, this tradeoff can be looked at as a major limitation. We have biologically realistic mathematical models of neurons but have no way to computationally model them such that the advantages they provide are actually exploited.
- **Access to Neuromorphic Hardware** - There are only a handful of quality neuromorphic devices currently in production, some of which probably aren't even accessible to the general public. Even the ones that are publically available aren't cheap and are both more difficult and less useful than cpu-based architectures.

Looking Ahead

- **CHIPS & Science Act**[17] - In August of 2022 our president signed into law the CHIPS and Science Act, which is our government's attempt to bolster U.S. leadership in semiconductors. The CHIPS and Science Act provides $52.7 billion for American semiconductor research, development, manufacturing, and workforce development.
- **Long-term solution to computational efficiency** - There is a direct relationship between the size of language models and the computational resources required to run them. SpikeGPT is able to implement a 260 billion parameter model, but that number will only increase in the future as models are trained on larger and more complex datasets. True to their nickname, next-generation neural networks, SNNs aren't going anywhere anytime soon. Their integration into the fold of application-based AI really is inevitable, they are too advantageous to be ignored.
- **Long-term solution to language comprehension** - As explained in the motivation section, our ability to understand and generate language relies on a very advanced, complex process distributed over multiple neurobiological systems in the brain. Excluding the benefit of computational efficiency, I think it stands to reason that neural networks that better capture the granular, biological characteristics of brain cells are better in the long run than neural networks that don't.

Work Cited

1. https://science.howstuffworks.com/life/evolution/language-evolve.htm#:~:text=It%20was%20first%20invented%20and,50%2C000%20and%20100%2C000%20years%20ago.
2. https://researchfeatures.com/understanding-mechanisms-language-comprehension/#:~:text=Language%20comprehension%20is%20one%20of,meanings%2C%20and%20general%20world%20knowledge.
3. https://www.credit-suisse.com/about-us-news/en/articles/news-and-expertise/chatgpt-the-potential-of-large-language-models-202303.html
4. https://gptblogs.com/chatgpt-how-much-data-is-used-in-the-training-process
5. https://www.datarobot.com/blog/introduction-to-dataset-augmentation-and-expansion/
6. https://en.wikipedia.org/wiki/Computational_neuroscience
7. https://en.wikipedia.org/wiki/Action_potential
8. https://home.csulb.edu/~cwallis/artificialn/History.htm
9. https://en.wikipedia.org/wiki/Hodgkin%E2%80%93Huxley_model
10. https://en.wikipedia.org/wiki/Biological_neuron_model
11. https://snntorch.readthedocs.io/en/latest/tutorials/tutorial_3.html
12. https://snntorch.readthedocs.io/en/latest/tutorials/tutorial_5.html
13. https://en.wikipedia.org/wiki/Spiking_neural_network
14. https://arxiv.org/pdf/2302.13939.pdf
15. https://johanwind.github.io/2023/03/23/rwkv_overview.html
16. https://www.researchgate.net/publication/320409442_On_Artificial_Spiking_Neural_Networks_Principles_Limitations_and_Potential
17. https://www.whitehouse.gov/briefing-room/statements-releases/2022/08/09/fact-sheet-chips-and-science(PDF) On Artificial Spiking Neural Networks: Principles, Limitations and Potential-act-will-lower-costs-create-jobs-strengthen-supply-chains-and-counter-china/