SpikingVTG: Saliency Feedback Gating Enabled Spiking Video Temporal Grounding

Malyaban Bal*
The Pennsylvania State University
University Park, PA 16802
mjb7906@psu.edu

Brian Matejek
SRI International
Arlington, USA
brian.matejek@sri.com

Susmit Jha
SRI International
Arlington, USA
susmit.jha@sri.com

Adam Cobb
SRI International
Arlington, USA
adam.cobb@sri.com

Abstract

Video Temporal Grounding (VTG) seeks to retrieve consecutive intervals or specific clips from a video based on specified natural language queries. VTG requires accurately aligning video segments with corresponding natural language instructions, highlighting the need for effective methodologies to capture semantic correspondence and maintain temporal coherence. Spiking neural networks (SNNs), previously underexplored in this domain, present a unique opportunity to tackle VTG challenges from both the architectural and energy-efficiency perspectives. In this paper, we leverage sparse spike-based communication of SNNs to propose a multimodal architecture tailored for VTG tasks, namely SpikingVTG, providing a biologically inspired and efficient solution. Leveraging temporal saliency feedback, our proposed spiking video-language model (VLM) achieves competitive performance with non-spiking VLMs across diverse moment retrieval and highlight detection tasks. We introduce a Saliency Feedback Gating (SFG) mechanism that improves performance while reducing overall neural activity. To efficiently train our spiking VLM, we analyze the convergence dynamics of each neuronal layer and utilize equilibrium states to enable training using implicit differentiation at equilibrium. This approach eliminates the need for computationally expensive backpropagation through time while also enabling the use of knowledge distillation for efficient model training. To further improve operational efficiency and facilitate the on-chip deployability of our model, we leverage a multi-stage training pipeline that focuses on eliminating non-local computations, such as softmax and layer normalization, leading to the development of the Normalization Free (NF)-SpikingVTG model. Additionally, we create an extremely quantized variant, a 1-bit NF-SpikingVTG model, which vastly improves computational efficiency during inference while maintaining minimal performance degradation from our base model. Our work introduces the first spiking model to demonstrate competitive performance on VTG benchmarks, including QVHighlights and Charades-STA.

1 Introduction

The rapid expansion of various social medias and portable smart technologies has triggered an unprecedented surge in video content. This vast influx of data has intensified the need for efficient

*Work done during internship at SRI International.

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

methods to retrieve and analyze video information. Consequently, the field of Video Temporal Grounding (VTG) ([1, 2]) has emerged as a vital area of research. The main objective of VTG is to identify the precise segment of a video that corresponds to a given natural language query, enabling more accurate and context-driven video content retrieval. In this paper, we focus on two tasks: moment retrieval ([3, 4]), which aims to identify video intervals relevant to a given query, and highlight detection ([5]), which retrieves the best candidate segment of the video in response to the query. Therefore, our work involves analyzing multimodal data—combining video content with natural language queries—to develop an effective solution to the problem. With the rise of foundation models like large language models, the field of VTG has seen significant advancements ([6, 1]). However, these models demand substantial computational power and energy ([7]) to operate. Furthermore, VTG is inherently resource-intensive, requiring the analysis of long video sequences, which leads to significant computational overhead. In this work, we leverage sparse spike-based communication and simplified accumulation-based computation in spiking neural networks (SNNs) to develop an efficient, lightweight solution for VTG ([8]).

Beyond the computational efficiency of SNN-based frameworks, we also harness their temporal dynamics to propose a spiking transformer-based VLM architecture (Fig. 1), namely **SpikingVTG**, that matches or surpasses the performance of current state-of-the-art non-spiking VLM. The input video for the VTG task typically consists of a long sequence of segments or clips. A key challenge in VTG is thus accurately identifying salient segments ([2]) or temporally dependent segments that exhibit a strong semantic correspondence with the given query. Our SNN-based VLM, operated over a period of simulation time steps, allows us to leverage its intermediate output as a feedback to identify the salient segments. We use the average spiking rate (ASR) of the output of the transformer-based SpikingVTG model to compute a dynamic saliency score of each video segment w.r.t the given query, which we then leverage as a mask for a multiplicative gating mechanism. This enables the model to focus on relevant portions of the video, as demonstrated in our experiments, while also reducing computational overhead by minimizing attention to irrelevant segments.

From a bio-plausibility perspective, as explored by Kar et al. [9], feedback based connection plays a prominant role in human visual cortex primarily responsible for object recognition. Furthermore, the feedback connection maintains the layer-wise convergence of ASR at equilibrium, enabling the implementation of an implicit differentiation framework ([10]), allowing for more efficient training of our model. This learning framework, leverages layer-wise converged ASR values at equilibrium to train the spiking model in one backpropagation step, instead of using the computationally expensive backpropagation through time (BPTT) ([11]). The SpikingVTG framework further involves a multi-stage training pipeline aimed at developing spiking models to fascilitate potential deployment on resource-constrained edge-based device enabled with neuromorphic chips. To allow for efficient training of our spiking model, we employ a knowledge distillation strategy ([12]), enabling knowledge transfer from a non-spiking UniVTG model, used as the "teacher", to our "student" SpikingVTG model. This process utilizes the ASR of converged intermediate states at equilibrium, enabling efficient training of our multimodal spiking video-language model.

Furthermore, transformer architectures ([13]) utilize non-local normalization operations such as softmax and layer normalization, which present challenges for implementation on neuromorphic hardware ([14]). To address this limitation, we introduce the Normalization-Free (NF)-SpikingVTG model, which eliminates all layer normalization operations and substitutes softmax spiking attention with a ReLU-based spiking attention mechanism. While ReLU-based attention mechanisms have previously been explored in non-spiking domains ([15]), we are the first to introduce this concept within the spiking attention mechanism, demonstrating competitive performance compared to softmax-based approaches. Additionally, to reduce computational complexity, following works on quantization in analog LLMs ([16]), we propose a 1-bit quantized variant of SpikingVTG. Our multi-stage training approach enables minimal performance degradation while enhancing operational efficiency during inference, in our SpikingVTG models. To our knowledge, this work is the first to evaluate the performance of an operational spiking VLM framework across various VTG tasks, including moment retrieval and highlight detection, on datasets such as QVHighlights and Charades-STA.

The primary contribution of our work are as follows:

- **SpikingVTG Model and Training Framework:** We propose a transformer-based, multi-modal spiking video language model with a spiking decoder module for moment retrieval and highlight detection in VTG tasks. We leverage the layer-wise convergence dynamics in

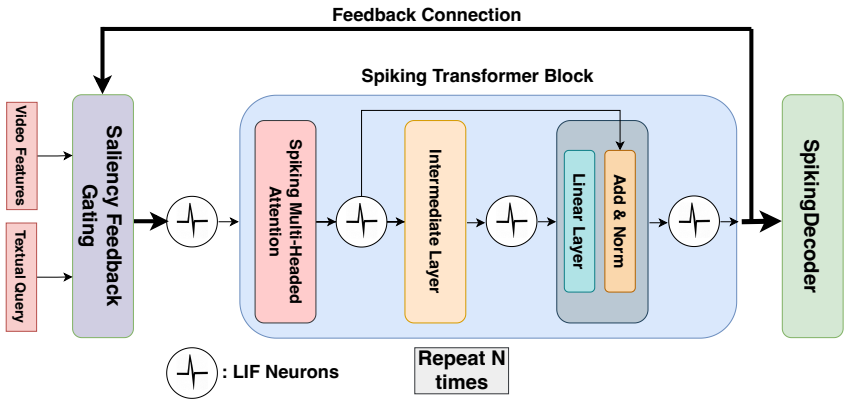

Figure 1: High-level overview of the proposed SpikingVTG architecture. The spiking Vision-Language Model (VLM) takes video and textual features as inputs, employing a spiking transformer core that utilizes Saliency Feedback Gating through temporal feedback connections. The model incorporates a spiking decoder module that takes the output of the transformer core to predict parameters for the VTG task.

our model to train our model using implicit differentiation at equilibrium, bypassing memory intensive BPTT. The result is the first spiking architecture to demonstrate competitive performance on VTG.

- **Saliency Feedback Gating Mechanism:** We introduce a saliency feedback gating mechanism for input video, that leverages the ASR of the output of the spiking transformer core at each time step. This temporal feedback enhances task-specific performance while minimizing neural activity, ultimately reducing overall computational overhead.
- **Multi-Stage Training Pipeline:** We propose a multi-stage training pipeline for our SpikingVTG framework, utilizing knowledge distillation and architectural modifications to create lightweight and computationally efficient spikingVTG variants. We replace computationally intensive non-local operations like layer normalization and softmax with hardware-friendly alternatives. We further introduce extreme quantization, developing a 1-bit NF-SpikingVTG model that significantly reduces memory as well as computational overhead.

2 Methodology

The VTG problem formulation is added in Appendix A. In this section we discuss the primary contributions of our work.

2.1 SpikingVTG: Architecture Overview

The core computational unit of the proposed SpikingVTG model is a leaky integrate-and-fire (LIF) neuron ([17]) detailed in Appendix B. Neurons communicate with each other using sparse, spike-based activations instead of real-valued signals, significantly improving energy/power efficiency. The model architecture includes a spiking transformer core for processing inputs, a saliency feedback gating mechanism for dynamic input control, and a spiking decoder module (Appendix **??**) to predict the parameters required for the VTG task, as described in Appendix A.

2.1.1 Spiking Transformer Core

The fundamental computational unit in our work is a leaky-integrate and fire neuron (LIF) (see Appendix A)The high-level overview of each encoder block of our spiking transformer architecture is demonstrated in Fig. 1. The model consists of N encoder layers, each consists of a spiking multi-headed attention block, followed by an intermediate layer and an output layer. Communication within and between encoder layers occurs via spikes. Furthermore, all matrix multiplications involved in linear layers and attention layer comprises of more efficient fp-accumulative (ACC) operations

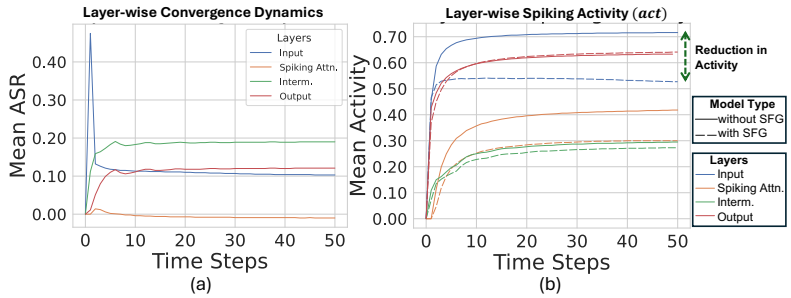

Figure 3: Results obtained upon passing a random input sample from QVHighlights dataset to our SpikingVTG models. (a) Graph shows convergence dynamics of layer-wise mean ASR against operating time steps for a randomly selected spiking transformer encoder layer (Fig. 1). It is to be noted that since we allow ternary spikes ASR can be negative as well. (b) Graph shows, layer-wise mean spiking activity ($act_i[t]$, averaged over number of neurons in that layer) against operating time steps in x-axis. The model with SFG shows markedly reduced activity in both the input layer and the spiking attention layer, underscoring its role in minimizing neuronal activity.

instead of fp-MAC operations in conventional neural architectures. Detailed descriptions of each layer are provided in the appendix. In the following section, we highlight modifications made to the architecture to facilitate the development of more efficient SpikingVTG variants. In Section E, we replace non-local normalization operations and introduce a ReLU-based attention mechanism. In Section F, we quantize all weights in the linear layers, including those in the intermediate and output layers, to 1-bit precision.

2.1.2 Saliency Feedback Gating (SFG)

SpikingVTG operates over a specific number of convergence time steps (T), with the convergence dynamics detailed in the following section. This temporal processing allows us to leverage intermediate outputs to dynamically update the input to the model at every time step for better predictions. This approach conforms to the feedback connections observed in the human visual cortex ([9]), providing a bio-plausible explanation for its efficacy. The average spiking rate of the final transformer encoder layer (a_N^t) of the Spiking Transformer module is used to compute a temporal saliency score with the input query enabling the design of a gating mechanism, allowing selective focusing on relevant segments of the video while minimizing computation on irrelevant segments. The operation in the saliency feedback gating mechanism (Fig. 2) is demonstrated below,

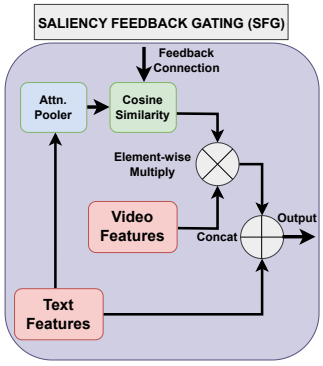

Figure 2: The internal operations of the saliency feedback gating mechanism.

$$
\begin{aligned}
F_s^{v_i}[t] = \cos(\mathbf{a_i^{N_v}}[\mathbf{t}], \mathbf{M}) := \frac{\mathbf{a_i^{N_v}}[\mathbf{t}] \cdot \mathbf{M}}{\|\mathbf{a_i^{N_v}}[\mathbf{t}]\|_2 \|\mathbf{M}\|_2}, \\
\bar{V}[t+1] = V * F_s^v[t] \\
I[t+1] = \bar{V}[t+1] \oplus \mathbf{Q}
\end{aligned}
\tag{1}
$$

where, using attentive pooling operation, sentence representation $\mathbf{M} = Softmax(\mathbf{W_p Q})\mathbf{Q}$, $\mathbf{M} \in \mathbb{R}^{1\times D}$, $\mathbf{Q} \in \mathbb{R}^{L_q \times D}$, $V \in \mathbb{R}^{L_v \times D}$ and $\mathbf{W_p}$ is a learnable embedding. $F_s^{v_i}[t]$ is the dynamic saliency score at time t for the segment of i of the video. $a_i^{N_v}[t]$ is the ASR of output of the spiking transformer for the the i corresponding segment of the video. The SFG mechanism not only results in better performance of our SpikingVTG architecture on evaluation metrics (see Table 2) but also reduces overall neural activity by sparsifying input spikes (Fig. 3b).

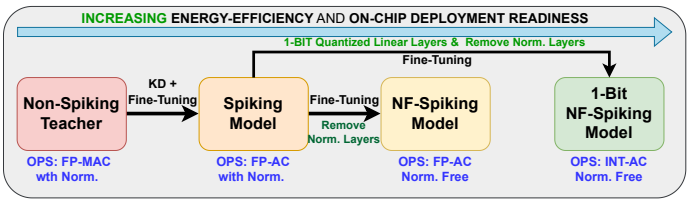

Figure 4: High-level overview of the multi-stage training framework for our proposed SpikingVTG models, enabling the development of lightweight and computationally efficient spiking models. Below each model we have noted the primary operations involved in that architecture.

2.2 Training Leveraging Convergence Dynamics

Following, Eqn. S1 & S3, we can formulate $a_i[t+1] = \frac{1}{V_{th_i}}(\hat{f}(a_{(i-1)}[t+1]) + b_i - \frac{u_i[t+1]}{\sum_{j=0}^{t}\gamma^j})$, where $\hat{f}$ is operation of layer i. As time approaches $t \to \infty$, the layer-wise ASRs converge to equilibrium, enabling the derivation of steady-state equations for linear layers ([10]). Moreover, surrogate steady-state functions can be formulated for non-linear layers ([18]) as,

$$a_i^* = \sigma(\frac{1}{V_{th_i}}(\hat{f}(a_{i-1}^*) + b_i)) \tag{2}$$

where, clipping function $\sigma(x)$ clamps the values within $[-1, 1]$. This is because following Eqn. S2, we allow ternary spikes thus ASR must be with $[-1, 1]$. The empirical convergence of ASR is shown in Fig. 3a. To analyze the overall layer-wise neural activity, which includes both positive and negative spiking event, we present the layer-wise dynamics of the absolute spiking events in Fig. 3b, i.e. $act_i[t] = \frac{\sum_{i=1}^{t}|s_i[t]|}{t}$. During training, leveraging implicit differentiation ([19]) at equilibrium, only ASR values at equilibrium are used,

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

where, θ is the model parameters, $g_\theta(a) = f_\theta(a) - a$, f is the steady-state equation of ASR, J^{-1} is the inverse Jacobian of g_θ when $a = a^*$, i.e., at equilibrium. Thus, unlike BPTT, we do not need to store the intermediate computational graph and the model parameters can be updated using a single backpropagation step.

2.3 Multi-Staged Training Pipeline

Training a multimodal spiking architecture like SpikingVTG is resource-intensive. To enhance the efficiency of this process and develop computationally efficient variants of our model, we propose a multi-staged training framework, as illustrated in Fig. 4. We utilize a non-spiking "teacher" VLM to guide the training of our "student" SpikingVTG model using knowledge distillation (see Appendix D). After this initial stage, we fine-tune SpikingVTG using the true labels. Once the base SpikingVTG model is established, we modify its architecture, as outlined in Appendix E and F, followed by additional fine-tuning to create computationally efficient variants with minimal performance degradation. The resulting computationally efficient, lightweight models are well-suited for deployment on neuromorphic chips, enabling efficient inference.

3 Experimentation

We evaluate all proposed spiking video-language models on moment retrieval and highlight detection tasks using the QVHighlights and Charades-STA datasets. Since, to the best of our knowledge, our proposed model is the first spiking VLM evaluated on VTG tasks, we benchmark its performance against state-of-the-art non-spiking video-language models. Additionally, we perform a study comparing our three model variants—SpikingVTG, NF-SpikingVTG, and 1-bit NF-SpikingVTG—on task specific performance and computational efficiency. Preliminary energy analysis further highlights the potential benefits of each model version.

Table 1: Performance comparison of SpikingVTG model with SFG against other non-spiking VTG solutions on the evaluation set of the QVHighlights and Charades-STA for moment retrieval task.

2*Method	SNN	QVHighlights				Charades-STA			
		@0.3	@0.5	@0.7	mAP@avg	@0.3	@0.5	@0.7	mIoU
UniVTG+PT ([2])	No	78.58	67.35	52.65	45.44	72.63	60.19	38.55	52.17
M-DETR ([1])	No	-	53.94	34.84	32.20	65.83	52.07	30.59	45.54
EaTR ([20])	No	-	61.36	45.79	41.74	-	-	-	-
2D-TAN ([3])	No	-	-	-	-	58.76	46.02	27.5	41.25
LLaViLo ([21])	No	-	-	-	-	-	55.72	33.43	-
UMT ([6])	No	-	60.26	44.26	38.59	-	49.35	26.16	-
QD-DETR ([22])	No	-	62.68	46.66	41.22	-	57.31	32.55	-
UniVTG ([2])	No	71.81	59.74	40.90	36.13	70.81	58.01	35.65	50.1
SpikeMba ([23])	No	-	65.32	51.33	44.84	71.24	59.65	36.12	51.74
SpikingVTG (Our Model)	**Yes**	**80.72**	**67.42**	**50.65**	**43.81**	**71.13**	**58.13**	**37.02**	**50.62**

3.1 Results

Our model outperforms non-spiking VTG models, including EaTR ([20]), 2D-TAN ([3]), M-DETR ([1]), LLaViLo ([21]), UMT ([6]), QD-DETR ([22]) and non-pretrained UniVTG model ([2]). Additionally, it achieves competitive results compared to the current state-of-the-art pretrained UniVTG model. It is important to note that SpikeMba ([23]) is not a fully spiking architecture; rather, one component of its multi-stage network uses an SNN. The results for each of the compared methods are taken from their respective papers. Our model establishes a baseline for future spiking VLM architectures on VTG tasks. The results are shown in Table 1 & S1.

Table 2: Comparing the performance of different SpikingVTG variants on the evaluation set of QVHighlights dataset.

2*Method	QVHighlights-MR				QVHighlights-HL		Activity
	@0.3	@0.5	@0.7	mAP@avg	@mAP	HIT@1	
SpikingVTG w/o SFG	78.65	65.10	47.46	42.56	40.60	67.42	0.41
SpikingVTG w/ SFG	**80.72**	**67.42**	**50.65**	**43.81**	**40.74**	**68.32**	0.34
NF-SpikingVTG w/ SFG	79.87	66.45	48.27	42.68	40.54	67.61	0.25
1-bit NF-SpikingVTG w/ SFG	78.77	65.16	47.35	42.32	40.31	67.29	**0.19**

3.2 Ablation Study

As shown in Table 2, the use of the Spike Feedback Gating (SFG) mechanism enhances performance compared to the model without SFG. Furthermore, as highlighted in Fig. 3 it results in reduced neuronal activity as well. Moreover, the computationally efficient NF-SpikingVTG model with SFG performs competitively even when compared to other state-of-the-art (SOTA) non-spiking video-language models (VLMs). Although the 1-bit NF-SpikingVTG variant shows a slight reduction in performance across evaluation metrics, it is highly memory efficient and involves simpler computations, making it well-suited for deployment on resource-constrained hardware. Furthermore, Table 2 also presents the average model-wide neural activity of the spiking model, calculated over $T = 12$ time steps. This metric represents the proportion of active neurons per timestep, averaged across all layers. This demonstrates that the optimizations aimed at enhancing computational efficiency (i.e. reducing non-local normalization operation and introducing quantized weights) also effectively reduce overall neural activity in the model. Preliminary energy analysis is in Appenix H.

4 Conclusions

Our saliency feedback gating-enabled SpikingVTG model offers a computationally efficient approach for VTG tasks while maintaining competitive performance with state-of-the-art non-spiking models. By harnessing layer-wise convergence dynamics, we efficiently train our model using implicit differentiation at equilibrium. We employ a multi-stage training pipeline that incorporates knowledge distillation, using the non-spiking pretrained UniVTG model as the "teacher" and the SpikingVTG model as the "student". This training pipeline further enables architectural optimizations, leading to the development of Normalization Free (NF)-SpikingVTG and 1-bit NF-SpikingVTG, enhancing computational efficiency and facilitating the on-chip deployment of these complex models.

5 Acknowledgments

This work was supported in part by the United States Air Force and Defense Advanced Research Projects Agency (DARPA) under Contract No. FA8750-23-C-0519, the Defense Advanced Research Projects Agency (DARPA) under Agreement No. HR0011-24-9-0424, and the U.S. Army Research Laboratory Cooperative Research Agreement W911NF-17-2-0196. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the Department of Defense or the United States Government.

References

[1] Jie Lei, Tamara L Berg, and Mohit Bansal. Detecting moments and highlights in videos via natural language queries. *Advances in Neural Information Processing Systems*, 34:11846–11858, 2021.

[2] Kevin Qinghong Lin, Pengchuan Zhang, Joya Chen, Shraman Pramanick, Difei Gao, Alex Jinpeng Wang, Rui Yan, and Mike Zheng Shou. Univtg: Towards unified video-language temporal grounding. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2794–2804, 2023.

[3] Songyang Zhang, Houwen Peng, Jianlong Fu, and Jiebo Luo. Learning 2d temporal adjacent networks for moment localization with natural language. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 12870–12877, 2020.

[4] Jonghwan Mun, Minsu Cho, and Bohyung Han. Local-global video-text interactions for temporal grounding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10810–10819, 2020.

[5] Fa-Ting Hong, Xuanteng Huang, Wei-Hong Li, and Wei-Shi Zheng. Mini-net: Multiple instance ranking network for video highlight detection. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XIII 16*, pages 345–360. Springer, 2020.

[6] Ye Liu, Siyuan Li, Yang Wu, Chang-Wen Chen, Ying Shan, and Xiaohu Qie. Umt: Unified multi-modal transformers for joint video moment retrieval and highlight detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3042–3051, 2022.

[7] Siddharth Samsi, Dan Zhao, Joseph McDonald, Baolin Li, Adam Michaleas, Michael Jones, William Bergeron, Jeremy Kepner, Devesh Tiwari, and Vijay Gadepally. From words to watts: Benchmarking the energy costs of large language model inference. In *2023 IEEE High Performance Extreme Computing Conference (HPEC)*, pages 1–9. IEEE, 2023.

[8] Samanwoy Ghosh-Dastidar and Hojjat Adeli. Spiking neural networks. *International journal of neural systems*, 19(04):295–308, 2009.

[9] Kohitij Kar, Jonas Kubilius, Kailyn Schmidt, Elias B Issa, and James J DiCarlo. Evidence that recurrent circuits are critical to the ventral stream's execution of core object recognition behavior. *Nature neuroscience*, 22(6):974–983, 2019.

[10] Mingqing Xiao, Qingyan Meng, Zongpeng Zhang, Yisen Wang, and Zhouchen Lin. Training feedback spiking neural networks by implicit differentiation on the equilibrium state. *Advances in Neural Information Processing Systems*, 34:14516–14528, 2021.

[11] Emre O Neftci, Hesham Mostafa, and Friedemann Zenke. Surrogate gradient learning in spiking neural networks: Bringing the power of gradient-based optimization to spiking neural networks. *IEEE Signal Processing Magazine*, 36(6):51–63, 2019.

[12] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

[13] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2017.

[14] Amar Shrestha, Haowen Fang, Zaidao Mei, Daniel Patrick Rider, Qing Wu, and Qinru Qiu. A survey on neuromorphic computing: Models and hardware. *IEEE Circuits and Systems Magazine*, 22(2):6–35, 2022.

[15] Kai Shen, Junliang Guo, Xu Tan, Siliang Tang, Rui Wang, and Jiang Bian. A study on relu and softmax in transformer. *arXiv preprint arXiv:2302.06461*, 2023.

[16] Hongyu Wang, Shuming Ma, Li Dong, Shaohan Huang, Huaijie Wang, Lingxiao Ma, Fan Yang, Ruiping Wang, Yi Wu, and Furu Wei. Bitnet: Scaling 1-bit transformers for large language models. *arXiv preprint arXiv:2310.11453*, 2023.

[17] Sangya Dutta, Vinay Kumar, Aditya Shukla, Nihar R Mohapatra, and Udayan Ganguly. Leaky integrate and fire neuron by charge-discharge dynamics in floating-body mosfet. *Scientific reports*, 7(1):8257, 2017.

[18] Malyaban Bal and Abhronil Sengupta. Spikingbert: Distilling bert to train spiking language models using implicit differentiation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 10998–11006, 2024.

[19] Shaojie Bai, J Zico Kolter, and Vladlen Koltun. Deep equilibrium models. *Advances in Neural Information Processing Systems*, 32, 2019.

[20] Jinhyun Jang, Jungin Park, Jin Kim, Hyeongjun Kwon, and Kwanghoon Sohn. Knowing where to focus: Event-aware transformer for video grounding. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 13846–13856, 2023.

[21] Kaijing Ma, Xianghao Zang, Zerun Feng, Han Fang, Chao Ban, Yuhan Wei, Zhongjiang He, Yongxiang Li, and Hao Sun. Llavilo: Boosting video moment retrieval via adapter-based multimodal modeling. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2798–2803, 2023.

[22] WonJun Moon, Sangeek Hyun, SangUk Park, Dongchan Park, and Jae-Pil Heo. Query-dependent video representation for moment retrieval and highlight detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 23023–23033, 2023.

[23] Wenrui Li, Xiaopeng Hong, and Xiaopeng Fan. Spikemba: Multi-modal spiking saliency mamba for temporal video grounding. *arXiv preprint arXiv:2404.01174*, 2024.

[24] Jan Hosang, Rodrigo Benenson, and Bernt Schiele. Learning non-maximum suppression. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4507–4515, 2017.

[25] Yufei Guo, Yuanpei Chen, Xiaode Liu, Weihang Peng, Yuhan Zhang, Xuhui Huang, and Zhe Ma. Ternary spike: Learning ternary spikes for spiking neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 12244–12252, 2024.

[26] Raphael Tang, Yao Lu, Linqing Liu, Lili Mou, Olga Vechtomova, and Jimmy Lin. Distilling task-specific knowledge from bert into simple neural networks. *arXiv preprint arXiv:1903.12136*, 2019.

[27] Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. Tinybert: Distilling bert for natural language understanding. *arXiv preprint arXiv:1909.10351*, 2019.

[28] Wu Liu, Tao Mei, Yongdong Zhang, Cherry Che, and Jiebo Luo. Multi-task deep visual-semantic embedding for video thumbnail selection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3707–3715, 2015.

[29] Jiyang Gao, Chen Sun, Zhenheng Yang, and Ram Nevatia. Tall: Temporal activity localization via language query. In *Proceedings of the IEEE international conference on computer vision*, pages 5267–5275, 2017.

[30] Song Han, Jeff Pool, John Tran, and William J. Dally. Learning both weights and connections for efficient neural networks, 2015.

A Video Temporal Grounding (VTG) Formulations

Given a video V and a language query Q, we first divide V into a sequence of L_v fixed-length clips $v_1, \ldots, v_{L_v}$, where each clip v_i is of length l and has a centered timestamp t_i. The free-form text query Q consists of L_q tokens, denoted as $Q = \{q_1, \ldots, q_{L_q}\}$. Following previous studies on VTG ([2]), we define three elements for each clip $v_i = (f_i, d_i, s_i)$, where $f_i = 1$ if the clip is in foreground, i.e. relevant else $f_i = 0$. $d_i = [d_{s_i}, d_{e_i}] \in \mathbb{R}^2$ represent the temporal distance that converts the clip timestamp t_i to its interval boundaries. Here, d_i is valid when $f_i = 1$. The term d_{s_i} denotes the distance between the start of the interval and t_i, while d_{e_i} denotes the distance between the end of the interval and t_i. Let $s_i \in [0, 1]$ be a continuous score that quantifies the relevance between the visual content of clip v_i and the query Q. Our proposed SpikingVTG model predicts these three parameters for each video clip. In this paper, we focus on specific VTG tasks, which are carried out as follows:

Moment Retrieval: We rank the predicted clip boundaries $\{\tilde{b}_i\}_{i=1}^{L_v}$, where $b_i = [t_i - d_{s_i}, t_i + d_{e_i}]$, based on their associated probabilities given by $\{\tilde{f}_i\}_{i=1}^{L_v}$. Since the predicted L_v boundaries are dense, we employ a 1-dimensional Non-Maximum Suppression (NMS) ([24]) with a threshold of 0.7 to eliminate highly overlapping boundary boxes, resulting in a final prediction.

Highlight Detection For each clip, we rank all clips based on their combined scores $\{\tilde{f}_i + \tilde{s}_i\}_{i=1}^{L_v}$. This combined value represents how well the chip i match with the underlying query. We then return the top clips (e.g., Top-1) as predictions.

B Spiking Neural Networks

The discrete time dynamics of an LIF-based spiking neuron can be given as follows,

$$\begin{aligned} u_i[t+\delta] &= \gamma u_i[t] + W_{(i-1)}(s_{(i-1)}[t]) + b_i, \\ u_i[t+1] &= u_i[t+\delta] - V_{th_i} s_i[t+1], \end{aligned} \tag{S1}$$

where, at time t, $u_i[t]$ is the membrane potential of the i^{th} neuronal layer; b_i indicates a bias term and γ is the leaky term. $W_{(i-1)}$ represents the layer-specific operation; $t + \delta$ is an intermediate time step to determine if the neuron fired; V_{th_i} is the threshold of layer i. We use a ternary spiking model ([25]) in our work for spike ($s[t+1]$) generation, thus the spiking operation is given as,

$$s_i[t+1] = \begin{cases} +1 & \text{if } u_i[t+\delta] \geq V_{th_i}, \\ -1 & \text{if } u_i[t+\delta] \leq -V_{th_i}, \\ 0 & \text{otherwise} \end{cases} \tag{S2}$$

This approach enhances performance while avoiding the introduction of additiional floating-point multiplicative and accumulative (fp-MAC) operations. The average spiking rate (ASR) [10] of LIF neurons within each layer i at time t can be defined as a weighted-average function:

$$a_i[t] = \frac{\sum_{\tau=1}^{t} \gamma^{t-\tau} s_i[\tau]}{\sum_{\tau=1}^{t} \gamma^{t-\tau}}. \tag{S3}$$

C Spiking Decoder

The spiking decoder comprises of stacked modules of 1-D convolution operations followed by integrate-fire (IF) neuron layers ($\gamma = 1$ in Eqn. S1), for spike generation. The decoder layer takes in the average spiking rate ($a_N^t \in \mathbb{R}^{L_v \times D}$) from the output of the final spiking transformer encoder block. The spiking decoder used for the foreground indicator (f_i) applies n_1 1-D convolution operations with kernel size k_1, each followed by an IF layer. The spiking decoder used for d_i applies n_2 1-D convolution operations with kernel size k_2, each followed by an IF layer, and the final convolution layer has 2 output channels to predict both components in d_i. The average spiking rate of individual decoders are used as predictions for the two underlying targets.

D Leveraging Knowledge Distillation (KD)

To enable efficient training of our spiking multimodal architecture, we utilize Knowledge Distillation (KD) techniques ([12, 26, 27]). We use the pre-trained UniVTG, currently the state-of-the-art in VTG, as a "teacher" and rather than distilling based on the prediction layer, we exploit the outputs of the internal layers of the "teacher". We establish a one-to-one mapping, by design, between the internal representations at equilibrium of our spiking transformer and the corresponding layers of the "teacher", ensuring that the number of layers in both architectures remains consistent. The internal representation-based KD is formulated as,

$$L_{KD} = \sum_{i=1}^{N} MSE(a^*_{r_i} W_d, T_{r_i}) \tag{S4}$$

where, $W_d \in \mathbb{R}^{d_s \times d_t}$ is a linear transformation that aligns the dimensionality of the layer of the "students" with that of the corresponding layer of the "teacher". $a^*_{r_i}$ denotes the converged ASR of the internal representation (dimension d_s) layer r_i, which is the output from the spiking transformer encoder layer i. T_{r_i} is the representation of the the corresponding block i (dimension d_t) of the teacher model. The KD process is an integral part of the framework and serves as the first stage of our multi-stage pipeline, followed by fine-tuning on the true labels (Fig. 4).

E Replacing Softmax and Removing Layer Normalization

In our work, we use a spiking attention mechanism (detailed in Appendix) which uses the key and value inputs as spikes instead of real values. Given d-dimensional queries, keys, and values $\{q_i[t], s_{k_i}[t], s_{v_i}[t]\}_{i=1}^{L}$, at time t, the attention weights α_{ij} are generally computed as follows:

$$\alpha_{ij}[t] = \phi\left(\frac{1}{\sqrt{d}}\begin{bmatrix} q_i[t]^\top s_{k_1}[t] & \cdots & q_i[t]^\top s_{k_L}[t] \end{bmatrix}\right)_j \tag{S5}$$

where, ϕ is usually the softmax function and output of spiking attention layer at time t is $attn_i[t] = \sum_{j=1}^{L} \alpha_{ij}[t] s_{v_j}[t]$. Given that softmax requires expensive non-local fp-MAC operations, we replace it with the less costly ReLU() operation and perform a simple scaling with L^{-1}. This replacement, while maintaining competitive model performance, is only feasible when following the multi-stage training process outlined in Fig. 4. This highlights the importance of the initial KD and fine-tuning stages, which help stabilize the model. Additionally, we explore the removal of all layer normalization layers, from Fig. 1, during training (as shown in Fig. 4), further streamlining the model design for on-chip deployment. We refer to the resulting model, which uses ReLU in place of Softmax and omits layer normalization, as a Normalization-Free (NF) spiking model.

F 1-BIT weight quantized SpikingVTG

Following prior work ([16]), 1-bit quantization consists of centering the weights W to achieve a zero mean, followed by binarization to $+1$ or -1 using the signum function as shown,

$$\begin{aligned} W_q &= \text{sgn}(W - \alpha), \\ \alpha &= \frac{1}{nm}\sum_{ij} W_{ij} \end{aligned} \tag{S6}$$

where, $W \in \mathbb{R}^{n \times m}$. The signum function, denoted as $\text{sgn}(x)$, categorizes the element x based on its sign. It outputs $+1$ when x is positive and -1 when x is zero or negative. . Additionally, the output of the linear layer is scaled by a constant $\beta = \frac{1}{nm}\sum_{ij} |W_{ij}|$. Thus, with ternary activations, our model now incorporates binary weights. Following the multi-stage learning approach illustrated in Fig. 4, our 1-bit SpikingVTG model emerges as a light-weight multimodal spiking VLM model. Additionally, we empirically demonstrate that employing binary weights while eliminating

normalization layers achieves competitive performance, resulting in 1-bit NF-SpikingVTG, enabling on-chip implementation and significantly improving computational efficiency. Thus, in the resulting model the primary computational operation involve interger accumulations since individual weight values are $W_{q_{ij}} \in \{-1, 1\}$ and activations are $s_{ij} \in \{-1, 0, 1\}$.

G Additional Experimental Details

The Spiking Transformer core in our model comprises four encoder layers, each with a hidden dimension of 1024, with 8 attention heads. For the knowledge distillation phase, we employ a pre-trained UniVTG model ([2] that has been fine-tuned on our specific dataset.

Dateset Details: QVHighlights ([1]) is the only public dataset that includes ground-truth annotations for moment retrieval and highlight detection, allowing for a thorough evaluation of the performance of our model and additional ablation studies. We also employ the Charades-STA dataset ([29]) to conduct further assessments on additional moment retrieval tasks.

Table S1: Performance comparison of our SpikingVTG model with SFG against other non-spiking VTG solutions on the evaluation set of the QVHighlights for highlight detection task.

2*Method	SNN	QVHighlights	
		mAP	HIT@1
UniVTG+PT ([2])	No	41.34	68.77
DVSE ([28])	No	18.75	21.79
XML+ ([1])	No	35.38	55.06
M-DETR ([1])	No	35.65	55.55
EaTR ([20])	No	37.15	58.65
M-DETR + PT ([1])	No	37.70	60.32
UniVTG ([2])	No	38.83	61.81
QD-DETR ([22])	No	39.13	63.03
UMT ([6])	No	39.85	-
SpikingVTG (Our Model)	**Yes**	**40.74**	**68.32**

Evaluation Metrics: For QVHighlights, following previous work ([1]) we use Recall@1 with IoU thresholds of 0.3, 0.5 and 0.7 and average mean average precision (mAP) as the evaluation metric for moment retrieval tasks. For highlight detection, we use mAP and HIT@1, where a clip is considered a true positive if it receives a score of "Very Good" ([6]). For Charades-STA, we employ Recall@1 with IoU thresholds of 0.3, 0.5, and 0.7, along with the mean IoU (mIoU).

H Analysis of Energy and Power Efficiency

We conduct a preliminary energy analysis of the proposed SpikingVTG variants during inference and compare it to a non-spiking UniVTG model with comparable depth and hidden state dimensions (also hidden dimension (D) is same as intermediate layer dimension in our implementation). From a simpler circuit design standpoint, for our analysis we consider 45nm CMOS technology and 32-bit precision, thus floating point (fp)-MAC operations consume $4.6pJ$, fp-ACC operations consume $0.9pJ$ and integer(int)-ACC operations consume $0.1pJ$ ([30]). The primary energy consumption is attributed to the transformer encoder layers, which consist primarily of the attention mechanism and multiple linear layers ([16]. The primary computation cost, calculated for an input sequence of length L, of each transformer encoder-layer of the non-spiking model can be expressed as: $Cost_A = [(3LD^2)+(LD^2+L^2D)+(LD^2)+(LD^2)]$ fp-MAC operations, corresponding to the three projection layers for query, key, and values, the attention mechanism, the intermediate layer, and the output layer.

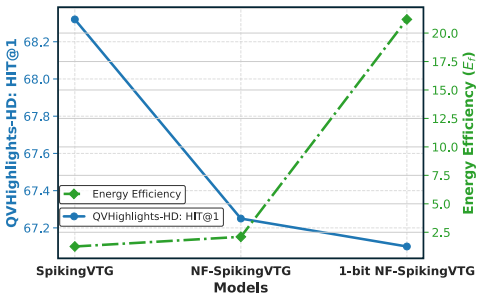

Figure S1: Graph depicting the performance of each model type on the QVHighlights highlight detection task, alongside their potential energy efficiency (E_f).

For the SpikingVTG model, per spiking transformer encoder layer the energy cost per time step is given by: $Cost_{S_t} = [(3 \cdot IFR_{in} \cdot LD^2) + (IFR_k \cdot LD^2 + IFR_v \cdot L^2D) + (IFR_{attn} \cdot LD^2) + (IFR_{interm.} \cdot LD^2)]$ fp-ACC operations, where each term is associated for each component similar to the one specified above. IFR_l represents the mean firing rate of the corresponding layer l. The total energy cost for the spiking model is: $Cost_S = (Cost_{S_t} * T)$ fp-ACC operations, where T represents the number of time steps. The models that include normalization also have an added cost of energy for normalization however, it is of the order $O(LD)$ so have not been included in our computation however it is to be noted that both our NF-SpikingVTG and 1-BIT NF-SpikingVTG

models are normalization free so they do not incur this added cost. Moreover, for 1-bit spikingVTG, the core computations in matrix multiplications shift from using fp-ACC to int-ACC operations.

We define energy efficiency of the spiking model as $E_f = Cost_A/Cost_S$. Specific examples of the energy analysis are provided in the Appendix. When operating the underlying models for $T = 12$ time steps, the energy efficiency and performance of each model are illustrated in Fig. S1. The average power efficiency for each model type is calculated as $P_f = E_f \times T$, demonstrating that our models are significantly more power-efficient (ranging from $15\times$ in SpikingVTG to upto $250\times$, in 1-bit NF-SpikingVTG) compared to non-spiking models. This efficiency arises from the ability of SNNs to unroll complex operations over time, thus providing low-powered solutions for complex tasks. Although this method of analysis does not account for various architectural energy advantages, it provides a useful approximation to gauge the potential benefits of spiking models over their non-spiking counterparts.