# Towards Energy-Preserving Natural Language Understanding With Spiking Neural Networks

Rong Xiao, Yu Wan, Baosong Yang, Haibo Zhang, Huajin Tang, *Senior Member, IEEE*, Derek F. Wong, *Senior Member, IEEE*, and Boxing Chen

***Abstract*—Artificial neural networks have shown promising results in a variety of natural language understanding (NLU) tasks. Despite their successes, conventional neural-based NLU models are criticized for high energy consumption, making them laborious to be widely applied in low-power electronics, such as smartphones and intelligent terminals. In this paper, we introduce a potential direction to alleviate this bottleneck by proposing a spiking encoder. The core of our model is bi-directional spiking neural network (SNN) which transforms numeric values into discrete spiking signals and replaces massive multiplications with much cheaper additive operations. We examine our model on sentiment classification and machine translation tasks. Experimental results reveal that our model achieves comparable classification and translation accuracy to advanced TRANSFORMER baseline, whereas significantly reduces the required computational energy to 0.82%.**

***Index Terms*—Natural language processing, language model, natural language understanding, spiking neural network.**

## I. INTRODUCTION

DEEP neural networks have shown their dominant performances across various natural language understanding (NLU) tasks in recent years [1], [2]. Generally, by transiting the discrete symbols to their corresponding continuous representations, neural-based NLP models can fully utilize the training examples to learn linguistic knowledge via semantic space mapping. In order to obtain outstanding performance, recent models increase their throughput with larger model size [3] or more hidden layers [4], [5]. Nevertheless, the high energy costs make neural NLU models fail to be widely employed in low-power electronics, *e.g.*, smartphones, and intelligent terminals [6]. It has become a widely known obstacle to the applicability of neural-based NLU systems.

The main reason for high energy consumption in vanilla neural models lies in a large amount of multiplication operations [7]. With the increase of dimensionality, multiplications in transformation functions exponentially enlarge the computational costs. To tackle this problem, existing studies mainly focus on compressing the hidden size via removing redundant dimensions, such as knowledge distillation [8] and channel pruning [9]. A natural problem arises: *Is it possible to replace multiplications with cheaper operations, thus optimizing the origin of high computational costs?*

After investigating the above potential direction for the energy reduction inside NLU models, the spiking neural network [SNN, [5]], caters to the demand of our model design. Specifically, SNN serves information as discrete spiking signals instead of continuous distributed representations, which mimics the mechanism inside the human brain. Concretely, the computational process inside SNN is mainly contributed by additive operations. Such important superiority of SNN can tremendously reduce the multiplications of decimal values, which cost far more energy than additive operations [10], [11]. Recent studies have proved that, SNN can be successfully deployed in computer vision and speech recognition [4], [12] applications. Nevertheless, its attempt at natural language processing tasks is rare to be explored.

In this paper, we propose a systematic spiking-based encoder, which stacks multiple bi-directional SNN layers to encode the input text into representations. Specifically, spiking signals are exploited with discrete activations to derive representations from trainable parameters. Also, to make the discrete function differentiable, we explore various strategies to approximate the spiking procedure for back-propagation. We examine the effectiveness of our method on sentiment analysis (*i.e.*, IMDb sentiment classification) and machine translation tasks (*i.e.*, IWSLT'15 English to Vietnamese (En-Vi) and WMT'17 Chinese to English (Zh-En)). Experimental results reveal that the proposed encoder is able to significantly reduce the energy cost to 0.82% of the strong TRANSFORMER baseline, in the meanwhile, marginally harming the performance of models engaging the proposed SNN encoder. Extensive analyses indicate that, our

Manuscript received 16 January 2022; revised 11 July 2022 and 12 September 2022; accepted 22 October 2022. Date of publication 10 November 2022; date of current version 9 December 2022. This work was supported in part by the National Key Research and Development Program of China, under Grant 2020AAA0105900, in part by the National Natural Science Foundation of China, under Grant 62206188, in part by the China Postdoctoral Science Foundation under Grant 2022M712237, in part by the Science and Technology Development Fund, Macau SAR under Grant 0070/2022/AMJ, in part by the Multi-year Research Grant from the University of Macau under Grant MYRG2020-00054-FST, in part by the National Key Research and Development Program of China under Grant 2018YFB1403202, and in part by the Alibaba Group through Alibaba Research Intern Program. Work was done when Rong Xiao and Yu Wan were interning at Damo Academy, Alibaba Group. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Nancy F. Chen. *(Rong Xiao and Yu Wan contributed equally to this work.) (Corresponding authors: Baosong Yang; Huajin Tang.)*

Rong Xiao, Baosong Yang, Haibo Zhang, and Boxing Chen are with the Damo Academy, Alibaba Group, Hangzhou 310000, China (e-mail: xiaorong.scu@gmail.com; nlp2ct.baosong@gmail.com; dreamfly.zhang@gmail.com; boxing.cbx@alibaba-inc.com).

Yu Wan and Derek F. Wong are with the NLP CT Lab, University of Macau, Macao 999078, China (e-mail: nlp2ct.ywan@gmail.com; derekfw@umac.mo).

Huajin Tang is with the Zhejiang University, Hangzhou 310027, China (e-mail: huajin.tang@gmail.com).

Digital Object Identifier 10.1109/TASLP.2022.3221011

approach outperforms existing binarized neural networks [13] and knowledge distillation-based compression approaches [8], [14] on both energy saving and model accuracy.

## II. Related Work

Recent neural networks rely on massive multiplication operations on float values during inference time. For example, TextCNN [15] uses cross-correlation operation to compute the similarity of two inputs, and Transformer [1] model conducts the scaled dot-product attention for alignments. Several model compression approaches have been proposed to reduce the computational complexity via decreasing the model throughput, *i.e.* dimensionality. He et al. [9] suggested removing redundant features in hidden states for eliminating useless calculations. Li et al. [13] proposed to binarize gates in the recurrent neural networks to accelerate the model inference. As a representative method, Hinton et al. [8] introduced a knowledge distillation (KD) scheme, which transfers useful information from a heavy teacher network to a portable student model. Jiao et al. [6] successfully exploited KD to pre-train a smaller language model for downstream tasks. Nevertheless, all those techniques compress existing models and restrict the model throughput, rather than directly solve the root cause of the problem – massive multiplications. In this paper, we aim to explore an efficient architecture that uses less computational complexity and lower energy consumption.

SNN is introduced to mimic the human brain by incorporating spikes into neural models [5]. Recent studies demonstrate that SNN is able to achieve promising performances and significantly reduce the energy consumption on object recognition, detection, and tracking tasks [12], [16]. For example, Kim et al. [12] propose Spiking-YOLO which applies the deep SNN to the object detection task. To expand the applicability of SNN, Yang et al. [16] introduce a hybrid paradigm – DashNet, to combine the advantages of vanilla neural network and SNN in a single model. As far as we know, those studies are mainly examined on computer vision and speech recognition tasks. Little work is arranged to explore the feasibility of SNN application on NLP tasks.

## III. Preliminary

### A. Transformer

Transformer [1] has shown great performance on various NLP tasks, such as machine translation [3] and language modeling [2]. Each layer in Transformer model consists of a multi-head self-attention network (MHSAN) and a feed-forward network (FFN). Specifically, given an input $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_t, \ldots, \mathbf{x}_l] \in \mathbb{R}^{l \times d}$ with sequence length and hidden size being $l$ and $d$, respectively, MHSAN determines how much information should be attended to the representation at each position[1]:

$$[\mathbf{Q}, \mathbf{K}, \mathbf{V}] = \mathbf{X}\left[\mathbf{W}_q, \mathbf{W}_k, \mathbf{W}_v\right], \tag{1}$$

$$\mathbf{O} = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d}}\right)\mathbf{V}, \tag{2}$$

$$\tilde{\mathbf{Y}} = LayerNorm\left(\mathbf{O}\mathbf{W}_o + \mathbf{X}\right), \tag{3}$$

where $\mathbf{W}_q, \mathbf{W}_k, \mathbf{W}_v \in \mathbb{R}^{d \times d}$ represents the learnable parameter matrices, $LayerNorm$ denotes the layer normalization.

FFN layer can collect semantic information via two linear transitions, connected with a non-linear ReLU activation:

$$\mathbf{Y} = LayerNorm\left(\mathbf{W}_2\left(\text{ReLU}\left(\mathbf{W}_1\tilde{\mathbf{Y}}\right)\right) + \tilde{\mathbf{Y}}\right), \tag{4}$$

where $\mathbf{W}_1 \in \mathbb{R}^{d \times 4d}$, $\mathbf{W}_2 \in \mathbb{R}^{4d \times d}$ are parameter matrices following Transformer setting [1].

### B. Gated Recurrent Units

As an alternative counterpart of MHSAN, recurrent neural network (RNN) recurrently captures the semantic information inside the input sequences. A variant of RNN, named gated recurrent unit [GRU, [17]], exploits a gate mechanism to simplify the computational process in long short-term memory [LSTM, [18]] model. Recent studies have shown that, when employing inside Transformer architecture [19], the machine translation model including bidirectional GRUs is able to yield comparable performance to that involving MHSANs on NLP tasks. Formally, the GRU recurrently captures the semantic information as:

$$\mathbf{r}_t = \sigma\left(\mathbf{W}_r\left[\mathbf{x}_t; \mathbf{h}_{t-1}\right]\right), \tag{5}$$

$$\mathbf{z}_t = \sigma\left(\mathbf{W}_z\left[\mathbf{x}_t; \mathbf{h}_{t-1}\right]\right), \tag{6}$$

$$\tilde{\mathbf{h}}_t = \tanh\left(\mathbf{W}_c\mathbf{x}_t + \mathbf{W}_p\mathbf{h}_{t-1} \otimes \mathbf{r}\right), \tag{7}$$

$$\mathbf{h}_t = \tilde{\mathbf{h}}_t \otimes (1 - \mathbf{z}) + \mathbf{h}_{t-1} \otimes \mathbf{z}. \tag{8}$$

where $\sigma$ indicates sigmoid function, $\otimes$ represents Hadamard product, $\mathbf{r}$ and $\mathbf{z}$ denote reset and update gate, respectively. $\{\mathbf{W}_r, \mathbf{W}_z\} \in \mathbb{R}^{d \times 2d}$ and $\{\mathbf{W}_c, \mathbf{W}_p\} \in \mathbb{R}^{d \times d}$ are trainable parameters. By calculating the gate value $\mathbf{r}$ and $\mathbf{z}$, the GRU model can flexibly control the balance of input representations, thus learning how much information should be derived from the previous hidden state $\mathbf{h}_{t-1}$ and current input $\mathbf{x}_t$.

### C. Spiking Neural Network

In an SNN, input data is typically represented as streams of spikes which are learned by spike-timing-based learning rules [20], [21]. The leaky integrate-and-fire [LIF, [22]] pattern was first proposed for the spiking neuron model, whose dynamics can evolve along with the temporal information according to the following equation:

$$\tau\frac{dV}{dt} = -(V - V_r) + R\left(I_o + I_{in} + I_n\right), \tag{9}$$

where $\tau$ represents the membrane time constant, $V$ is the membrane potential and $V_r$ is the rest, $R$ is the membrane resistance, $I_o$ is the constant inject current, $I_n$ is a background noise current which is set to 0, and $I_{in}$ is the input current. When $V$ exceeds a constant threshold $V_{thr}$, the neuron is set to fire, and $V$ is reset

[1]We exploit multi-head mechanism and bias terms in our implementation while omitting it in equations for simplification.

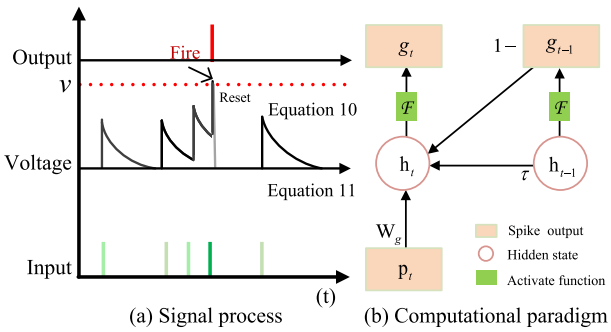



Fig. 1. Illustration of (a) signal process in SNN and (b) the computational paradigm of SNN cell. The input data in SNN is represented as streams of spikes. The model accumulates spikes to voltage. When the potential reaches a certain threshold $\nu$, it fires a spike and is reset to 0. For the voltage (i.e. hidden state $h_t$) at $t$-th step, the update equation is the sum of the input tensor at present and the hidden state at $t-1$ step. $\tau$ represents the membrane time constant and $\mathcal{F}$ denotes the spiking-convert activate function.

to $V_r = 0$. For ensuring computational tractability, this can be reimplemented with a more explicitly iterative manner [23]:

$$V^t = \left(1 - \frac{dt}{\tau}\right) V^{t-1} + \frac{dt}{\tau} I_{in}, \tag{10}$$

where $(1 - \frac{dt}{\tau})$ is the decay factor, expressed as $\tau$. The input current is the linearing summation $\sum_j W_j g(j)$, where $j$ indicates the index of pre-synapse and $g(j)$ is the input from presynaptic neuron $j$ at time $t$ which is one or zero, and $w_j$ is the weight of the $j$-th neuron:

$$V^t = \tau V^{t-1} + \sum_j W_j g(j). \tag{11}$$

By integrating the firing-and-resetting (FAR) mechanism, the final update equations are described below:

$$V_n^t(i) = \tau V_n^{t-1}(i)\left(1 - g_n^{t-1}(i)\right) + \sum_j W_{ij} g_{n-1}^t(j) + b, \tag{12}$$

$$g_n^{t-1} = \mathcal{F}\left(V_n^{t-1} - V_{thr}\right), \tag{13}$$

where $n$ represents the $n$-th layer, and $W_{ij}$ is the synaptic weight from the $j$-th pre-synaptic neuron to the $i$-th post-synaptic neuron. $\mathcal{F}$ is the spiking-convert function which gives 0 if input $x < 0$ otherwise 1. In this way, the model is able to propagate information regardless of multiplication operations, as shown in Fig. 1.

## IV. Spiking Encoder

### A. Bidirectional Spiking Layer for NLU

In an SNN, input data is typically represented as streams of spikes which are computed by spike-timing-based learning rules [5], [24]. The leaky integrate-and-fire [LIF, [25]] pattern is introduced for the spiking neuron model. It accumulates spikes to membrane potential. When the voltage reaches a certain threshold, the model transfers its information by firing a spike to the next neuron.

Specifically, our spiking encoder model first transforms the $t$-th input representation $\mathbf{x}_t$ to spikes $\mathbf{p}_t$ following the spiking-convert function $\mathcal{F}$:

$$\mathbf{p}_t = \mathcal{F}\left(\mathbf{x}_t - \nu\right), \tag{14}$$

where $\nu$ represents the activation threshold. For the $i$-th dimension, the spiking-convert function $\mathcal{F}(\mathbf{k}^i)$ gives 0 if input $\mathbf{k}^i$ is lower than 0 otherwise 1 as shown in Fig. 2(b). In this paper, we set $\nu = 0.0$ as default. For capturing the contextual information, we assign spikes $\mathbf{g}_{t-1}$ to take the features of the last time step $\mathbf{h}_{n-1}^t$ into account:

$$\mathbf{g}_{t-1} = \mathcal{F}\left(\mathbf{h}_{t-1} - \nu\right), \tag{15}$$

In order to measure the computational tractability, we integrate the firing-and-resetting [FAR, [23]] to balance the signals from the input and previous hidden state. Accordingly, the current accumulated voltage (hidden state) $\mathbf{h}_t$ can be formally expressed as:

$$\mathbf{u}_t = \mathcal{G}\left(\tau \mathbf{h}_{t-1}, 1 - \mathbf{g}_{t-1}\right), \tag{16}$$

$$\mathbf{v}_t = \sum_{i=1}^{d} \mathcal{G}\left(\mathbf{W}_{i,\cdot}, \mathbf{p}_t\right), \tag{17}$$

$$\mathbf{h}_t = \mathbf{u}_t + \mathbf{v}_t, \tag{18}$$

where $\mathbf{W} \in \mathcal{R}^{d \times d}$ indicates the trainable parameters. A decay factor $\tau \in (0, 1)$ is used to shrink the contribution from previous hidden states. Here, we set $\tau = 0.5$ as default. $\mathcal{G}(\mathbf{x}, \mathbf{s})$ uses flags where $\mathbf{s}^i$ is equal to 1 to derive the $i$-th representation or value from $\mathbf{x}$, otherwise the corresponding output is padded by zero. In this way, spiking signals can derive representations from parameters $\mathbf{W}$ without linear transformation, thus reducing the computational complexity.

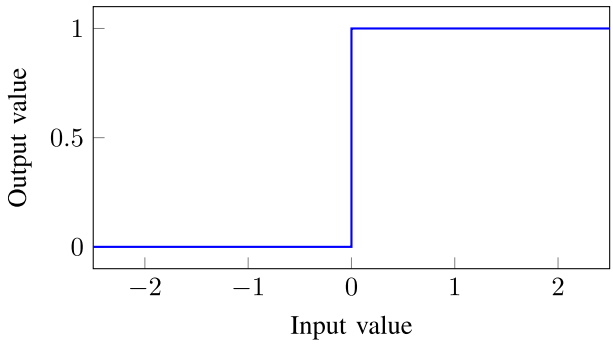
Similar to bi-directional RNN which considers both the forward and backward contexts, we propose bi-directional SNN (bi-SNN), where every single bi-SNN layer contains two SNN sublayers. Specifically, one sublayer collects the representations from the beginning to the end and another follows a reverse direction. After collecting hidden states generated by two sublayers, the hidden states are concatenated as the output representation of the SNN layer. We use the superscripts $b$ and $f$ for denoting the backward and forward directions:

$$\mathbf{Y} = \left[\mathbf{h}_t^f; \mathbf{h}_t^b\right]. \tag{19}$$

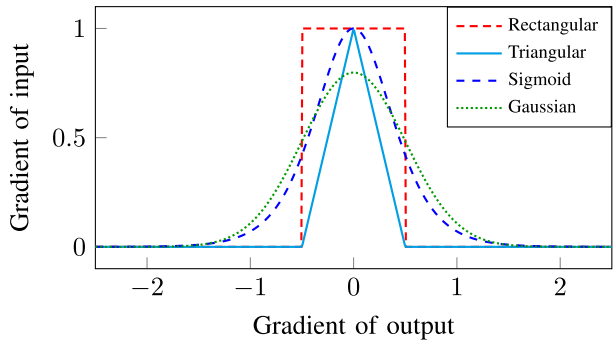
Interestingly, the major difference between SNN and RNN lies in the connection pattern and activation function [26]. For the neurons in the same layer, SNN prevents each neuron from connecting with other neurons, while RNN allows it. Moreover, the activation function of SNN is essentially a step function with discrete outputs, whereas RNN includes non-linear functions (*e.g.*, hyperbolic tangent) as activations.

### B. Updating Through Back-Propagation

Since the output of LIF activation is discrete and differentiable (Fig. 2(a)), it fails to immediately exploit the back-propagation




(a) The forward calculation of SNN cell.




(b) Gradient back-propagation patterns in SNN cell.

Fig. 2. The forward calculation and backward propagation in an SNN cell. In this case, we determine the active threshold as 0. The output of SNN cell is discrete and fail to cope with differentiable calculation. To simulate the gradient during back-propagation, we conclude 4 approximate functions, namely Rectangular, Triangular, Sigmoid and Gaussian.

mechanism to update parameters. An alternative way is to predefine continual derivatives of back-propagation patterns. Inspired by [24], we introduce 4 patterns to approximate the derivative of spike activity as shown in Fig. 2(b). Given the input $\mathbf{h}_t$ and its corresponding spiking output $\mathbf{g}_t$, the gradient of input $\nabla \mathbf{h}_{n-1}^t$ can be calculated by that of output $\nabla \mathbf{p}_t$ as follows:

- Rectangular: The gradient value is the same within the restricted region:

$$\nabla \mathbf{h}_t = \begin{cases} 1 & \text{if } \nabla \mathbf{p}_t \in [-0.5, 0.5), \\ 0 & \text{otherwise.} \end{cases} \tag{20}$$

- Triangular: The calculation is formed with two linear functions with the maximum value being 1:

$$\nabla \mathbf{h}_t = \begin{cases} 2\nabla \mathbf{p}_t + 1 & \text{if } \nabla \mathbf{p}_t \in [-0.5, 0), \\ -2\nabla \mathbf{p}_t + 1 & \text{if } \nabla \mathbf{p}_t \in [0, 0.5], \\ 0 & \text{otherwise.} \end{cases} \tag{21}$$

- Sigmoid: The gradient calculation is smoothed via sigmoid function $\sigma$:

$$\nabla \mathbf{h}_t = 4\sigma\left(4\nabla \mathbf{p}_t\right)\left(1 - \sigma\left(4\nabla \mathbf{p}_t\right)\right). \tag{22}$$

- Gaussian: We use a modified Gaussian function for back-propagation:

$$\nabla \mathbf{h}_t = \sqrt{\frac{2}{\pi}} e^{-2\nabla \mathbf{p}_t^2}. \tag{23}$$

For subsequent experiments, we empirically employ the Gaussian function as default. The comparison of different models is discussed in Section 7.1.

## V. Energy Consumption Comparison

One of the most outstanding advantages of applying SNN into the model is the lower energy cost. This can be demonstrated from (18), that the computational operations within SNN cell at each time step consists of **far less multiplicative operations** than that of MHSAN (1)–(3), FFN layer (4), as well as GRU (5)–(8). To fully compare the energy cost of different NLU models, we conclude the number of required operations in each submodule following Deng et al. [27], which only contains addition (add) and multiplication (mul) operators:

- The linear transformation from a vector with $d$ dimensions to another with $d$ dimensions yields $d^2$ additive and $d^2$ multiplicative operations, respectively;
- The non-linear activations (e.g. sigmoid and hyperbolic tangent functions) are implemented within polynomial numerical operations. This entry can be omitted compared to linear translation [27].
- As to numeric comparative and memory linking operations, the related process is arranged inside specific units. Compared to additive/multiplication operations, the activation function and representation selection operations inside SNN cost very little energy, which is ignored in common practice [11], [28], [29], [30], [31].

TABLE I
Statistics on Computational Operations, Number of Trainable Parameters, and Energy Cost of Typical NLU Components

| module | # add | # mul | energy($pJ$) | # params. | ratio (%) |
|---|---|---|---|---|---|
| MHSAN | $4d^2$ | $4d^2$ | $18.4d^2$ | $4d^2$ | 100.00 |
| FFN | $8d^2$ | $8d^2$ | $36.8d^2$ | $8d^2$ | 200.00 |
| CNN | $kd^2$ | $kd^2$ | $4.6kd^2$ | $kd^2$ | 75.00 |
| GRU | $6d^2$ | $6d^2$ | $27.6d^2$ | $6d^2$ | 150.00 |
| SNN | $d^2\rho$ | $d\rho$ | $3.7d\rho + 0.9d^2\rho$ | $d^2$ | **2.47** |

$d$ represents the model dimensionality, $k$ is kernel size in CNN with 3 as default, and $\rho$ indicates the spiking ratio with 0.5 as default. Note here that, for a large value of $d$, the additive entries with non-highest polynomial terms can be omitted. For simplicity, we only keep the highest term composing of $d$ for further calculation. Statistics are concluded upon an input sequence with the length being 1.

Based on these rules, we conclude the number of computational operations and the theoretical energy cost of typical NLU components in Table I . When calculating self-attention weights in MHSAN, the number of additive and multiplicative operations for linear transformation in (1) and (3) are $4d^2$. Note that, for a large value of $d$, the additive entries with non-highest polynomial terms can be omitted, such as the layer normalization module and softmax function. We can further compute the number of operations for the FFN layer as $8d^2$ according to (4). Considering the GRU model (5)–(8), we can learn that, the numbers of

additive and multiplicative operations are both $6d^2$ based on the different linear transformations. Moreover, we compute the number of add/mul operation following Rathi and Roy [32] for linear transformation of convolutional layer. With kernel size being $k$, the numbers of additive and multiplicative operations are both $kd^2$, respectively. Since the spiking signals (0 or 1) in the SNN cell can be used as masks to derive corresponding representations, only additive operations ($d\rho$) are considered in linear transformation. Meanwhile, considering the decay factor, the number of multiplicative operations is $d^2\rho$, where $\rho$ means whether a neuron will fire a spike. Referred to [7] [7], the energy consumption of add/mul operator are approximately 0.9/3.7 *pJ* ($pJ = 10^{-12}$ Joule). Accordingly, we calculate the energy cost ratio within each module. Firstly, the total energy cost of the MHSAN model is set as 100% (baseline system). Compared to the MHSAN module, FFN, CNN, and GRU modules require 2.0, 0.75, and 1.5 time(s) energy consumption, respectively. Importantly, the energy cost of a single SNN layer is far less than conventional neural models, with a ratio of approximately 2.47% of MHSAN, which is far lower than other modules.

## VI. Experiments

To examine the effectiveness of our method, we compare our spiking encoder with several typical NLU modules on two typical tasks, i.e. sentiment analysis and machine translation. As to the baselines for comparison:

- Transformer: We choose Transformer [1] as one of our baseline systems, as it has been the dominant model architecture across NLP tasks;
- TextCNN: Kim [15] used convolutional layer to classify the sentiment of reviews. We involve this approach as one of our baseline systems;
- bi-GRU: We also examine the effectiveness of bi-directional GRU layers [17] by incorporating them into Transformer architecture by replacing MHSAN components.
- Synthesizer-Dense: Especially, for machine translation tasks, we choose the dense version of Synthesizer Transformer [33], where the attention logits inside MHSANs are learnable parameters instead of dot-product calculation using query and key representations.

### A. Sentiment Analysis

*1) Task Specification and Dataset:* The sentiment analysis task aims at judging the sentiment of each text. By receiving the input text, the model is required to determine whether the corresponding text is positive or negative. We choose Internet Movie Database [IMDb, [34]] to verify the performance of our model. IMDb dataset contains customer reviews of movies, and includes 25,000 labeled training reviews and 25,000 labeled test reviews.

*2) Parameter Setting:* For a fair comparison, we set all the baseline implementations and our proposed model by using 3 network layers, and the dimensionalities of word embedding are all 100 [35]. Especially, for Transformer baseline, the number of heads in the MHSAN layer is 4, and the dimensionality of the inner connection in the FFN layers is 4 times of embedding size. For the bi-SNN model, the sizes of hidden state and word embedding are all 100, and the max pooling is used to derive representations along with the dimension of sequence length, thus obtaining a $d$-dimensional contextual semantics for prediction. During training, we choose the learning rate at 0.001, and the dropout ratio at 0.5. We set the decay factor as 0.2, and the threshold for firing as 0.5. We initialize word embeddings with the pre-trained GloVe [36] word vectors[2] across all models. Each experimental result represents the averaged value and variance of prediction accuracy over 5 independent experimental runs.

TABLE II
Classification Accuracy, Energy Consumption, and Storage Space of Models on IMDb Task

| Model | Accuracy (%) | Energy (%) | Storage (%) |
|---|---|---|---|
| Transformer | **89.70 ± 0.45** | 100.00 | 100.00 |
| TextCNN | 89.40 ± 0.52 | 50.00 | 196.16 |
| bi-GRU | 88.10 ± 0.48 | 50.00 | 100.02 |
| bi-SNN | 87.20 ± 0.49 | **0.82** | **3.26** |

As seen, the proposed spiking encoder can achieve 87.20% accuracy. Meanwhile, it only requires 0.82% energy comsumption, and 3.26% storage space against TRANSFORMER baseline.

*3) Model Performance:* Experimental results are concluded in Table II. As seen, Transformer outperforms all other baseline systems, with around 0.3% and 1.6% improvement over TextCNN and bi-GRU approach, respectively. Our proposed bi-SNN model shows marginal loss against the baseline. Most importantly, compared to Transformer baseline model, our bi-SNN model only uses 0.82% energy consumption, indicating the effectiveness of our model on energy saving.

*4) Storage Space:* The first layer of SNN converts word embeddings into spikes. The values in the spiking embedding matrix are binarized, each of which only requires 1 b for storage. In comparison, one FP32 value requires 32 bits. Consequently, replacing the word embedding matrix with its spiking signals can reduce the storage consumption to $\frac{1}{32}$ of origin. This provides an alternative solution to compress the storage space of the NLU model at inference time. Following this idea, we convert the word embeddings into spikes after training to collect the storage space. As in Table II, results show that our bi-SNN model only uses 3.26% storage space compared with Transformer baseline model.

### B. Machine Translation

*1) Task Specification and Dataset:* The neural machine translation (NMT) task aims at building a model that can translate a sentence from the source language to the target side, where the encoder is in charge of language understanding at the source side, and the decoder can take the contextual representations into account to translate accurate sentence at the target side. We choose three MT tasks, i.e. IWSLT'15 English-Vietnamese (En-Vi) and WMT'17 Chinese-English (Zh-En), containing

[2]https://nlp.stanford.edu/projects/glove/glove.6B.zip

TABLE III
BLEU SCORE (%) UPON TEST SET ON IWSLT'15 EN-VI AND WMT'17 ZH-EN MACHINE TRANSLATION TASKS AND ENERGY CONSUMPTION OF EACH MODEL SETTING

| Model | IWSLT'15 | WMT'17 | Energy (%) |
|---|---|---|---|
| TRANSFORMER | 30.62 ± 0.07 | **23.78 ± 0.08** | 100.00 |
| bi-GRU | **30.98 ± 0.08** | 23.70 ± 0.05 | 50.00 |
| SYNTHESIZER-Dense | 30.26 ± 0.03 | 22.54 ± 0.04 | 87.50 |
| bi-SNN | 30.53 ± 0.07 | 22.92 ± 0.08 | **0.82** |

The energy cost comparison is conducted on the encoder of each translation model. Compared to existing approaches, our spiking encoder achieves 0.82% and 1.64% energy consumption against TRANSFORMER and bi-GRU encoder, respectively.

13.3 k and 20.1 M training examples, respectively. All datasets are tokenized and turecased with mosesdecoder toolkit[3], and regulated into subword units [37] with 32 k byte-pair encoding (BPE) merging steps.

*2) Parameter Setting:* We use the TRANSFORMER-*base* setting as default across all models. The decoder of all models is the same as the conventional decoder from TRANSFORMER. Following Vaswani et al. [1], we set the number of layers in each encoder to 6, and the number of heads in the MHSAN module to 8. The dimensionalities of word embedding and the inner connection of FFN are 512 and 2,048, respectively. The hidden sizes of bi-GRU and bi-SNN in the encoder layer are all 512. During model training, we use the same learning rate schedule, which linearly increases at the warm-up phase, and then decays with respect to the squared root of the number of overpassed steps. The number of warm-up steps is 8k, and the maximum learning rate is 0.0007 [3].

*3) Model Performance:* As shown in Table III, our TRANSFORMER baseline system achieve better translation quality to the reported results in existing studies [38], making our evaluation convincing. Moreover, 6 bi-GRU encoder layers can achieve comparable results against TRANSFORMER model which is identical with previous findings [19]. Although the proposed SNN-based encoder shows around 0.5–0.8 BLEU score drops over different tasks, it greatly reduces the required computational energy to 0.82% compared with the baseline.

## VII. ANALYSIS

### A. Impact of Hyper-Parameters

A decay factor simulating information forgetting is crucial for obtaining representation which presents the accumulative semantics till the current step. Moreover, the spiking thresholds can effectively control the input representations and hidden states, so as to filter the important information for processing at the present step. To this end, we first investigate how the decay factor $\tau$ and spiking threshold $\nu$ influence the model performance. As seen in Tables IV and V, a proper decay factor $\tau$ and spiking threshold $\nu$ can guide the SNN model learning in a proper way, otherwise, the model easily diverges. $\tau = 0.5$ and $\nu = 0.0$ yield best performance than other settings. It is interesting to see that when $\tau = 0.0$, the proposed methods fail to converge, since the SNN cell is a token-wise spiking activation without any attendance to forward or backward contexts.

[3]https://github.com/moses-smt/mosesdecoder/

TABLE IV
COMPARISON OF SNN MODELS WITH DIFFERENT DECAY FACTOR $\tau$ ON EN-VI TASK. N/A: FAIL TO TRAIN

| $\tau$ | 0.00 | 0.25 | 0.5 | 0.75 | 1.00 |
|---|---|---|---|---|---|
| BLEU (%) | N/A | 28.81 | **30.53** | 28.77 | N/A |

TABLE V
COMPARISON OF SNN MODELS WITH DIFFERENT THRESHOLD $\nu$ ON EN-VI TASK. N/A: FAIL TO TRAIN

| $\nu$ | -1.0 | -0.5 | 0.0 | 0.5 | 1.0 |
|---|---|---|---|---|---|
| BLEU (%) | N/A | 29.21 | **30.53** | 29.16 | N/A |

TABLE VI
PERFORMANCE OF SNN MODELS WITH DIFFERENT BACK-PROPAGATION PATTERN ON EN-VI TASK

| Back-Propagation Pattern | BLEU (%) |
|---|---|
| Rectangular | 29.81 |
| Triangular | 30.09 |
| Sigmoid | 30.31 |
| Gaussian | **30.53** |

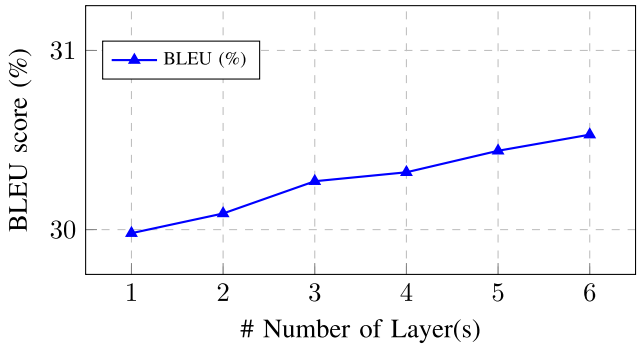



Fig. 3. BLEU score upon En-Vi task with respect to the number of bi-SNN layers.

*1) Back-Propagation Pattern:* SNN model can be implemented with multiple variations due to the back-propagation pattern, decay factor, and spiking threshold. Thus we further conduct a series of experiments to testify the performance of each back-propagation pattern. Results are concluded in Table VI. As seen, the Gaussian and Sigmoid functions give relatively better BLEU scores than other patterns. We conclude to the reason, that these two patterns simulate the gradient propagation in a continuous space, whereas Rectangular and Triangular patterns contain discrete values.

*2) Impact of the Number of Layers:* We further test our model with the different numbers of SNN layers. The results are reported in Fig. 3. We conclude that the translation quality increases gradually with the increase of network depth. This indicates that the proposed spiking encoder has potentialities to be further enhanced via expanding model size.

TABLE VII
COMPARISON BETWEEN BI-SNN AND BINARIZED BI-GRU ON EN-VI TASK

| Encoder Model | BLEU (%) |
|---|---|
| bi-SNN | **30.53** |
| binarized bi-GRU | 28.45 |
| binarized bi-GRU w/ finetuning | 30.29 |

Our approach not only has the ability to save energy, but also yields slightly better performance than binarized gated GRU.

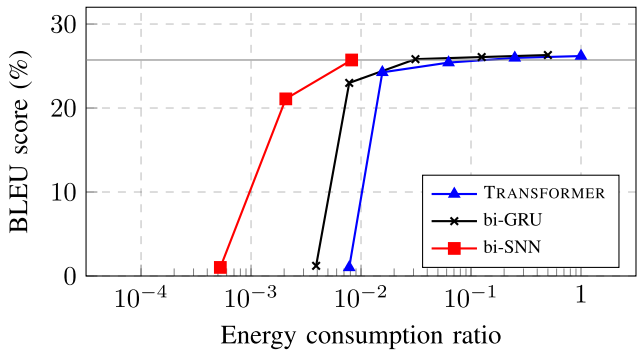



Fig. 4. Performance of variant models with respect to the ratio of energy consumption on En-Vi task. We regard the energy consumption of TRANSFORMER baseline as 1. We control the energy consumption of each model by modifying the dimensionality of hidden size. Moreover, knowledge distillation is leveraged for the compression of baseline models. We accumulatively halve the dimensionality of model till untrainable (from 512 to 32). Obviously, our model has the better capability to reduce the energy costs than its model compression counterparts.

### B. Comparison to Related Studies

*1) SNN Vs. Binarized Gated GRU:* A similar approach to our method is binarized gated GRU [13] which binarizes continual values in gates of GRU for speeding up the model inference. We give the performance comparison between the bi-SNN model and the binarized bi-GRU model over the En-Vi task. For the binarized bi-GRU model, we use two conversion methods: a) In the inference phase, we directly map the original values into 0 or 1 according to the specific threshold (set to 0); and b) We train the binarized bi-GRU model following [13]. Experimental results demonstrate that our model achieves slightly better performance than its binarized bi-GRU counterpart. Most importantly, our approach is able to reduce energy consumption, while binarized GRU is not since it still preserves product operations of the conventional GRU model.

*2) SNN Vs. Knowledge Distillation:* In order to investigate the relation between consumed energy and model performance, we further conduct a group of experiments on the dimensionality of the encoder. Here, we introduce the knowledge distillation [8] scheme to keep the performance of baseline systems following Tang et al. [14]. In Fig. 4, we simulate the energy consumption of each model with modified dimensionality $d$, and conduct the relationship between corresponding energy cost and performance. As seen, by accumulatively halving the dimensionality of model $d$ from 512, TRANSFORMER and bi-GRU model significantly lose the performance, and finally diverge when $d$ is 32. In comparison, the proposed spiking encoder can save almost 15 times more energy than that of baselines with knowledge distillation, verifying the superiority of our approach.

## VIII. CONCLUSION

In this paper, we introduce a spiking encoder for language understanding. We design our proposed spiking encoder with multiple SNN layers, and enhance each layer with better adaptation and back-propagation patterns. After conducting the experimental results on several NLP tasks, Compared to conventional baselines, our spiking encoder not only reveals comparable performances, but also significantly reduces the required energy cost. Our contributions are mainly in the following:

- We introduce a potential direction to reduce the energy cost from the perspective of computational operation rather than the model compression;
- We present a spiking encoder for NLU tasks that exploits additive operations to replace massive multiplications in conventional neural-based models;
- Our experiments on sentiment analysis and machine translation reveal that our model can reduce the energy cost to 0.82% against the advanced TRANSFORMER model.

Compared with existing advanced NLP research, the following directions related to our proposed spiking encoder raise our interest the most. First, as the big success of recent NLP approaches, the pre-trained language models (*e.g.*, BERT [2] and XLM-R [39]) have shown dominant performance. We believe that, for the large-scale spiking encoder, its performance and training method can be explored in the future. Besides, our spiking encoder shows its performance drop on the large-scale dataset setting. Considering this, a better trade-off between performance and energy cost is also worth being investigated. Last, SNN shows its own priority on specific devices. The energy cost of the spiking encoder can also be further examined following this setting. We wish this research can attract more studies to pay attention to the topic of "green NLP," and explore advanced techniques on low-energy NLP models in the future.

## ACKNOWLEDGMENT

The authors would like to thank the reviewer and action editor for reviewing our paper. Their suggestions are truly helpful for us to refine our manuscript.

## REFERENCES

[1] A. Vaswani et al., "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5998–6008.
[2] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics: Hum. Lang. Technol.*, 2019, pp. 4171–4186.
[3] M. Ott, S. Edunov, D. Grangier, and M. Auli, "Scaling neural machine translation," in *Proc. Workshop Statist. Mach. Trans.*, 2018, pp. 1–9.
[4] J. Wu, Y. Chua, M. Zhang, Q. Yang, G. Li, and H. Li, "Deep spiking neural network with spike count based learning rule," in *Proc. IEEE Int. Joint Conf. Neural Netw.*, 2019, pp. 1–6.
[5] A. Tavanaei, M. Ghodrati, S. R. Kheradpisheh, T. Masquelier, and A. Maida, "Deep learning in spiking neural networks," *Neural Netw.*, 2019,vol. 111, pp. 47–63.

[6] X. Q. Jiao et al., "TinyBERT: Distilling BERT for natural language understanding," in *Proc. Findings, Empirical Methods Natural Lang. Process.*, 2020, pp. 4163–4174.
[7] M. Horowitz, "1.1 Computing's energy problem (and what we can do about it)," in *Proc. IEEE Int. Solid-State Circuits Conf. Dig. Tech. Papers*, 2014, pp. 10–14.
[8] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," 2015, *arXiv:1503.02531*.
[9] Y. He, X. Zhang, and J. Sun, "Channel pruning for accelerating very deep neural networks," in *Proc. Int. Conf. Comput. Vis.*, 2017, pp. 1389–1397.
[10] H. Chen et al., "AdderNet: Do we really need multiplications in deep learning?," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 1468–1477.
[11] Y. Wan et al., "Attention mechanism with energy-friendly operations," in *Proc. Findings, Assoc. Comput. Linguistics*, 2022, pp. 3969–3976.
[12] S. Kim, S. Park, B. Na, and S. Yoon, "Spiking-YOLO: Spiking neural network for real-time object detection," *Proc. AAAI Conf. Artif. Intell.*, vol. 34, no. 07, pp. 11270–11277, 2020.
[13] Z. Li et al., "Towards binary-valued gates for robust LSTM training," in *Proc. Int. Conf. Mach. Learn.*, 2018, vol. 80, pp. 3001–3010.
[14] R. Tang, Y. Lu, L. Liu, L. Mou, O. Vechtomova, and J. Lin, "Distilling task-specific knowledge from bert into simple neural networks," 2019, *arXiv:1903.12136*.
[15] Y. Kim, "Convolutional neural networks for sentence classification," in *Proc. Empirical Methods Natural Lang. Process.*, 2014, pp. 1746–1751.
[16] Z. Yang et al., "DashNet: A hybrid artificial and spiking neural network for high-speed object tracking," 2019, *arXiv:1909.12942*.
[17] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," in *Proc. Neural Inf. Process. Syst. 2014 Workshop On Deep Learn.*, 2020.
[18] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.
[19] M. X. Chen et al., "The best of both worlds: Combining recent advances in neural machine translation," in *Proc. Assoc. Comput. Linguistics*, 2019, pp. 76–86.
[20] R. Gütig and H. Sompolinsky, "The tempotron: A neuron that learns spike timing–based decisions," *Nature Neurosci.*, vol. 9, no. 3, pp. 420–428, 2006.
[21] Q. Yu, R. Yan, H. Tang, K. C. Tan, and H. Li, "A spiking neural network system for robust sequence recognition," *IEEE Trans. Neural Netw. and Learn. Syst.*, vol. 27, no. 3, pp. 621–635, Mar. 2016.
[22] W. Gerstner and W. M. Kistler, *Spiking Neuron Models: Single Neurons, Populations, Plasticity*. Cambridge, U.K.: Cambridge Univ. Press, 2002.
[23] Y. Wu, L. Deng, G. Li, J. Zhu, Y. Xie, and L. Shi, "Direct training for spiking neural networks: Faster, larger, better," in *Proc. Assoc. Adv. Artif. Intell. Conf. Artif. Intell.*, 2019, vol. 33, pp. 1311–1318.
[24] Y. Wu, L. Deng, G. Li, J. Zhu, and L. Shi, "Spatio-temporal backpropagation for training high-performance spiking neural networks," *Front. Neurosci.*, vol. 12, 2018, Art. no. 331.
[25] A. Taherkhani, A. Belatreche, Y. Li, G. Cosma, L. P. Maguire, and T. M. McGinnity, "A review of learning in biologically plausible spiking neural networks," *Neural Netw.*, vol. 122, pp. 253–272, 2020.
[26] W. He et al., "Comparing SNNs and RNNs on neuromorphic vision datasets: Similarities and differences," *Neural Netw.*, vol. 132, pp. 108–120, 2020.
[27] L. Deng et al., "Rethinking the performance comparison between SNNs and ANNs," *Neural Netw.*, vol. 121, pp. 294–307, 2020.
[28] F. Akopyan et al., "TrueNorth: Design and tool flow of a 65 mW 1 million neuron programmable neurosynaptic chip," *Proc. IEEE Trans. Comput.-Aided Des. Integr.*, vol. 34, no. 10, pp. 1537–1557, 2015.
[29] N. P. Jouppi et al., "In-datacenter performance analysis of a tensor processing unit," in *Proc. 44th Annu. Int. Symp. Comput. Architecture*, 2017, pp. 1–12.
[30] C. Frenkel, M. Lefebvre, J.-D. Legat, and D. Bol, "A 0.086-mm$^2$ 12.7-pJ/SOP 64k-synapse 256-neuron online-learning digital spiking neuromorphic processor in 28-nm CMOS," *IEEE Trans. Biomed. Circuits Syst.*, vol. 13, no. 1, pp. 145–158, Feb. 2018.
[31] L. Liang et al., "H2learn: High-efficiency learning accelerator for high-accuracy spiking neural networks," in *Proc. IEEE Trans. Comput.-Aided Des. Integrated*, 2021, pp. 4782–4796.
[32] N. Rathi and K. Roy, "DIET-SNN: Direct input encoding with leakage and threshold optimization in deep spiking neural networks," 2020, *arXiv:2008.03658*.
[33] Y. Tay, D. Bahri, D. Metzler, D.-C. Juan, Z. Zhao, and C. Zheng, "Synthesizer: Rethinking self-attention for transformer models," in *Proc. Int. Conf. Mach. Learn.*, 2021, pp. 10183–10192.
[34] A. L. Maas, R. E. Daly, P. T. Pham, D. Huang, A. Y. Ng, and C. Potts, "Learning word vectors for sentiment analysis," in *Proc. 49th Annu. Meeting Assoc. Comput. Linguistics: Hum. Lang. Technol.*, 2011, pp. 142–150.
[35] A. Zhang, Z. C. Lipton, M. Li, and A. J. Smola, "Dive into deep learning," 2020. [Online] Available: https://d2l.ai,
[36] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *Proc. Empirical Methods Natural Lang. Process.*, 2014, pp. 1532–1543.
[37] R. Sennrich, B. Haddow, and A. Birch, "Neural machine translation of rare words with subword units," in *Proc. Assoc. Comput. Linguistics*, 2016, pp. 1715–1725.
[38] Y. Wan et al., "Self-Paced learning for neural machine translation," in *Proc. Empirical Methods Natural Lang. Process.*, 2020, pp. 1074–1080.
[39] A. Conneau et al., "Unsupervised cross-lingual representation learning at scale," in *Proc. Assoc. Comput. Linguistics*, 2020, pp. 8440–8451.

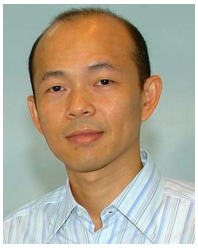
**Rong Xiao** received the bachelor's and Ph.D. degrees from the College of Computer Science, Sichuan University, Chengdu, China, in 2016 and 2021, respectively. She is currently an Assistant Professor with the College of Computer Science, Sichuan University. Her research interests include spike-based coding and learning for spiking neural networks and its application in neural morphological audiovisual perception.

**Yu Wan** is currently working toward the Ph.D. degree with the NLP$^2$CT Laboratory, University of Macau, Zhuhai. He has been receiving advice from Prof. Derek F. Wong since 2018. Since May 2020, he has been a Research Intern with DAMO Academy, Alibaba Group, advised by Dr. Baosong Yang. His research interests include machine learning, machine translation, translation quality estimation, and translation metric.

**Baosong Yang** received the Ph.D. degree from the NLP$^2$CT Laboratory of the University of Macau, Zhuhai, in 2019, advised by Prof. Derek F. Wong. He is currently an Algorithm Expert with the Language Technology Laboratory with Alibaba DAMO Academy. His research mainly include machine learning and natural language processing, especially multilingual NLP.

**Haibo Zhang** received the master's degree from the Institute Computing Technology, Chinese Academy of Sciences, Beijing, China, in 2014, advised by Prof. Qun Liu. He is currently a Principal Engineer with Language Service Department with Shopee. His research mainly include natural language processing, especially machine translation.

**Huajin Tang** (Senior Member, IEEE) received the B.Eng. degree from Zhejiang University, Hangzhou, China, in 1998, the M.Eng. degree from Shanghai Jiao Tong University, Shanghai, China, in 2001, and the Ph.D. degree from the National University of Singapore, Singapore, in 2005. He is currently a Professor with the College of Computer Science and Technology, Zhejiang University. His research interests include neuromorphic computing and robotic cognition. Dr. Tang was the recipient of the 2016 IEEE Outstanding TNNLS Paper Award and the 2019 IEEE Computational Intelligence Magazine Outstanding Paper Award. He was an Associate Editor for the IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS, Frontiers in *Neuromorphic Engineering*, and *Neural Networks*. He is the EIC of IEEE TRANSACTIONS ON COGNITIVE AND DEVELOPMENTAL SYSTEMS, and also a Board of Governor Member of the International Neural Networks Society.



**Derek F. Wong** (Senior Member, IEEE) received the Ph.D. degree in automation from Tsinghua University, Beijing, China, in 2005. He is currently an Associate Professor with the Department of Computer and Information Science, University of Macau, Zhuhai. His research interests include natural language processing and machine translation. He is the Leader of the Natural Language Processing and Portuguese–Chinese Machine Translation (NLP$^2$CT) Research Group and the Founder of the NLP$^2$CT Laboratory.

**Boxing Chen** is currently a Senior Staff Algorithm Expert with the Machine Intelligence Labratory of Alibaba Group. He works on natural language processing and Machine Learning. Prior to Alibaba, he was a Research Officer with the National Research Council Canada. He has coauthored more than 90 papers with the NLP/Speech conferences and journals. He was the recipient of the The Best Paper Award from MT Summit 2013, and The Best Paper Award Nomination from ACL 2013. He was the Area Chair for ACL, EMNLP and AACL. His teams ranked first place more than 20 times in various machine translation competitions.