

# FS-SS: Few-Shot Learning for Fast and Accurate Spike Sorting of High-channel Count Probes

Tao Fang<sup>1</sup> and Majid Zamani<sup>1</sup>

<sup>1</sup>Affiliation not available

March 24, 2025

# FS-SS: Few-Shot Learning for Fast and Accurate Spike Sorting of High-channel Count Probes

Tao Fang, *Student Member, IEEE*, Majid Zamani, *Member, IEEE*

**Abstract:** There is a need for fast adaptation in spike sorting algorithms to implement brain-machine interface (BMIs) in different applications. Learning and adapting the functionality of the sorting process in real-time can significantly improve the performance. However, deep neural networks (DNNs) depend on large amounts of data for training models and their performance sustainability decreases when data is limited. Inspired by meta-learning, this paper proposes a few-shot spike sorting framework (FS-SS) with variable network model size that requires minimal learning training and supervision. The framework is not only compatible with few-shot adaptations, but also it uses attention mechanism and dilated convolutional neural networks. This allows scaling the network parameters to learn the important features of spike signals and to quickly generalize the learning ability to new spike waveforms in recording channels after few observations. The FS-SS was evaluated by using freely accessible datasets, also compared with the other state-of-the-art algorithms. The average classification accuracy of the proposed method is 99.28%, which shows extreme robustness to background noise and similarity of the spike waveforms. When the number of training samples is reduced by 90%, the parameter scale is reduced by 68.2%, while the accuracy only decreased by 0.55%. The paper also visualizes the model's attention distribution under spike sorting tasks of different difficulty levels. The attention distribution results show that the proposed model has clear interpretability and high robustness.

**Index Terms**—Implantable brain-machine interface (iBMI), unsupervised spike sorting, attention mechanism, dilated convolutional neural network, few-shot (FS) learning, meta-learning, fast channel adaptation, robust feature extraction.

## I. INTRODUCTION

Extracellular recordings have been widely used to monitor neuronal activity by implanting multi-electrodes in the cortex and capturing multidimensional neural data. A processing step, known as spike sorting shown in Fig.1, is necessary to separate the multi-unit neural activities and assign the captured spikes to their originating neurons [1-6]. Spike sorting is an invaluable research tool applied in implantable brain-machine interface (iBMI) research for studying and decoding neural signals from different brain regions and

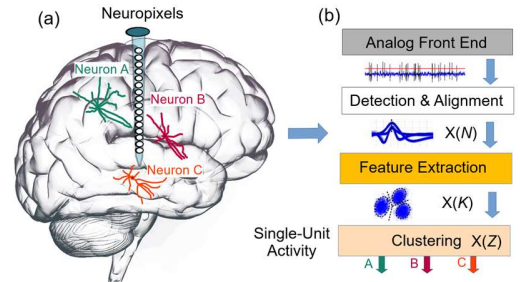


Fig. 1. (a) Recording activities from neurons A, B and C using Neuropixels electrode arrays. (b) Spike sorting process for determining single unit activity. Data is progressively reduced over processing blocks ( $Z < K < N$ ).

understanding the mechanisms of the brain. It is extremely beneficial in design and development of various applications such as identifying the optimal patterns and parameters to condition diseases by artificially modulating irregular or faulty electrical impulses [7], realizing a communication bridge for control of assistive devices for patients with damaged sensory/motor functions such as hand prosthesis [8], and stimulating a particular pathway for biological functionality regularization [9].

The recent trend in brain sensing is about the utilization of high-channel count neural interfaces that include tens of thousands of sensing probes [10]. There are two important aspects associated with the large-scale data streaming:

(i) *Recording channel variability*: three factors are involved:

a) neurons can show different dynamics based on electrode location. (b) neurons have various structures depending on the cortex regions and layers [11] which results in generation of unique actions potentials. And (c) action potentials are also function of time [12]. Considering such variations, there is high probability of spike shape changes over time. Therefore, a solid model is required to adapt itself to the recording channel variations.

(ii) *Training time*: The second challenge is training time which means adaptation to the characteristics of the recording channel. This certainly requires huge amount of training data as the deep learning algorithms are data hungry [13].

There is a need for a spike classification algorithm that adapts (or learns), and embeds intermittently the information about

T. F. Author was with Engineering and Applied Technology Research Institute, Fudan University, 220 Handan Road, Shanghai, 200433, China (e-mail: 19210860039@fudan.edu.cn).

M. Z. Author is with the School of Electronics and Computer Science, University of Southampton, Southampton, SO17 1BJ UK, (e-mail: m.zamani@soton.ac.uk).

new channel with only a few observations through a fast learning approach.

In recent years, different types of DNNs have been developed and utilized in spike sorting [14]. However, few-shot learning is not widely adopted in spike sorting. For example, Wu et al. [15] in 2019 developed a semi-supervised spike classification method called few-shot spike sorting using general adversarial network (GAN). The authors in [15] successfully categorized spikes within the utilized synthetic data: using 50 labelled spikes from a recording with a noise standard deviation of 5%, an accuracy of 97.7% was achieved. By increasing the number of labelled spikes to 200, the accuracy improved to 98.2%. In the presence of noise with a standard deviation of 20%, they reached an accuracy level of 66.4% (50 labelled spikes) and 85.5% (200 labelled spikes) [15]. Also tested on experimental data, it was demonstrated that training the GAN on particular waveforms with only a “few shots” was sufficient to recognize similar spikes in inference mode.

Few-shot learning is also used in other biomedical signal processing applications. In [16], authors proposed a novel framework referred to as the few-shot hand gesture recognition (FS-HGR). The FS-HGR quickly adapts by using a small number of observations to a new gesture (or user) through learning on its prior experience. Tested on second Ninapro databases (also referred to as the DB2) [17], which consists of 50 gestures (rest included) from 40 healthy subjects, FS-HGR led to 85.94% classification accuracy. FS-HGR led to 83.99% classification accuracy on new repetitions with few-shot observations (5-way 5-shot). Also, authors in [16] achieved 81.29% accuracy on new subjects with few-shot observation (5-way 5-shot). In [18], authors introduced a few-shot learning (FSL) applicability for electrocardiogram (ECG) signal proximity-based classification. They embedded the FLS to a deep convolutional neural network to recognize 2, 5 and 20 different heart disease classes. The extracted QRS complexes from PTB-XL dataset [19] containing the labelled 10-second raw signal ECG was used to train the model in [16]. The FSL-based classification provides continuous classification accuracy augmentation without network adjustments, and it achieves classification of healthy and sick patients ranging from 93.2% to 89.2%.

In essence, a solid and structured study on few-shot learning for spike sorting application is required, as having access to an implantable processor for changing its parameters is an extremely challenging and tedious task. This paper introduces a few-shot learning framework for improved spike classification accuracy. Compared with traditional methods, the introduced framework only requires a small number of training data to achieve high-performance deep learning spike sorting. The main contributions of this paper are summarized as follows:

- The FS-SS framework alleviates the problem of few observations learning in the spike sorting where the number of training samples are extremely limited.
- The FS-SS model dynamically adjusts the number of output channels in the convolution layer and the dropout rate according to the number of training data. This ensures optimal resource utilization to minimize computational overhead and hardware resources.

- Unique building blocks are designed and introduced in FS-SS. Embedding module to integrate few-shot adaptations capability, residual attention (RA) module to enhance classification robustness in noisy conditions and residual dilated convolution (RDC) to capture neuronal activity patterns at different scales and improve the model's ability to process multi-scale features while maintaining suitable resolution and reducing computational complexity.

To the best of our knowledge, this is the first time that a paper outlines in detail a few-shot adaptation algorithm in spike sorting. The rest of the paper is structured as follows: Section II describes the proposed architecture, including details of the individual constituent modules and how these modules are organized in the FS-SS framework. Section III describes the used datasets for performance analysis and the evaluation methods. Section IV scrutinizes the performance of the FS-SS framework by comparing it with other sorting methods, testing adaptation with different dataset proportions and visualization of sorting and attention results. Finally, Section V makes some concluding remarks.

## II. PROPOSED SPIKE SORTING ARCHITECTURE

This section first briefly explains the meta-learning method for the few-shot problem, then describes the basic building blocks of the proposed FS-SS network, and finally illustrates the overall FS-SS architecture and how to align the meta-learning method with the spike sorting process.

### A. Meta-learning Problem

In recent years, meta-learning [20] has been offering an extremely effective approach in solving problems that require minimal supervision [21-23]. Meta-learning is also called learning to learn aiming to improve the adaptability of the model to new tasks by learning the distribution of multiple tasks, rather than relying solely on the data of a single task [24]. This learning approach enables the deep learning algorithms to quickly adapt to new tasks by training on multiple different but related tasks. It is extremely suitable for classification problems with high complexity, high dimensions, and few samples such as spike sorting.

Unlike traditional supervised learning methods, the minimum data for updating neural network parameters based on meta-learning is a task rather than a single spike sample [25]. A task contains multiple spikes, including support ( $S$ ) spikes and query ( $Q$ ) spikes, so a task consists of  $S + Q$  spikes. The support set and query set are defined as in Eq. (1) and (2), where  $x_i^j$  represents the  $j^{\text{th}}$  sample of the  $i^{\text{th}}$  class and  $y_i^j$  is its corresponding label. The query set usually has only one sample per class, so the superscript  $j$  is omitted.

$$S = \{(x_i^j, y_i^j)\}_{i=1, j=1}^{N, k} \quad (1)$$

$$Q = \{(x_i, y_i)\}_{i=1}^q \quad (2)$$

The meta-learning problem with  $N$  classes and  $k$  samples in each category is called  $N$ -way  $k$ -shot problem [25] as shown in Fig. 2. In the proposed FS-SS framework, a 2-way 2-shot meta-learning is adopted which means that the support set has 2 classes ( $N=2$ ) each class has 2 samples ( $k=2$ ) with one sample

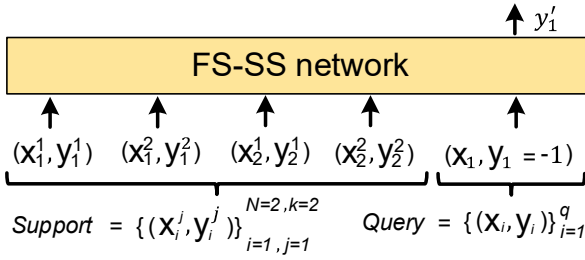


Fig.2. Meta-learning-based FS-SS network training process. The input includes support spikes and query spikes. In this paper, the support set contains two classes ( $N=2$ ), each class includes two spikes ( $k=2$ ), and the query set contains only one spike ( $q=1$ ). The FS-SS network is used to predict the label corresponding to the query set spike.

in the query set ( $q=1$ ). Therefore, a total number of 5 spike waveforms, which can be expressed as follows:

$$S = \{(x_1^1, y_1^1), (x_1^2, y_1^2), (x_2^1, y_2^1), (x_2^2, y_2^2)\} \quad (3)$$

$$Q = \{(x_1, y_1)\} \quad (4)$$

Inspired by [26], the true labels of the support set are fed into the network while the labels of the query set are treated as non-existent labels represented by  $-1$  in the training phase. Then these  $N \times k + 1$  labels and their corresponding sequences  $X$  are fed into the network. In the parameter update phase, the loss value  $\mathcal{L}_Q(f_\theta)$  is calculated only for the output label of the query set  $f_\theta(x_q)$  and the true label  $y_q$  as indicated in Eq. (5). This loss is used to update the learnable parameters  $\theta$  of the entire network and obtain the new parameters  $\theta'$ . Where  $\alpha$  represents the learning rate, and  $\nabla_\theta$  represents the result of the gradient of  $\theta$  as indicated in Eq. (6). After multiple iterations of the above process, the model will learn how to encode the support set and predict the true label of the query set [26].

$$\mathcal{L}_Q(f_\theta) = \text{loss}(f_\theta(x_q), y_q) \quad (5)$$

$$\theta' = \theta - \alpha \nabla_\theta \mathcal{L}_Q(f_\theta) \quad (6)$$

### B. The Building Modules of the FS-SS Framework

This section explains the design ideas and implementation details of the constituent modules in the FS-SS framework. It finally outlines the overall architecture of the FS-SS.

#### 1) The Embedding Module

The embedding block integrates the few-shot observations capability onto the FS-SS network. It performs preliminary feature extraction and to map input spike waveforms (5 spikes, 66 samples per spike) onto a standardized feature vector [27]. As shown in Fig.3, the embedding block first extracts the primary time domain features of the spike signal through a 1D convolutional layer. The kernel size in the 1D convolutional layer is set to 3, it has stride of 1 and also padding size is set to 1. The number of output channels shown by  $f$  in the 1D convolutional layer (i.e., the number of convolution kernels) is not fixed and it depends on the number of the spikes used in the training dataset. When the training dataset is large, the number of convolution kernels can be dynamically increased to capture wider features. On the other hand, when the training dataset is small, the number of convolution kernels can be appropriately reduced to avoid overfitting. This design ensures that the model can dynamically adjust its complexity according to the size of the dataset, thereby achieving a balance between extracting rich features and preventing overfitting. In addition, smaller number

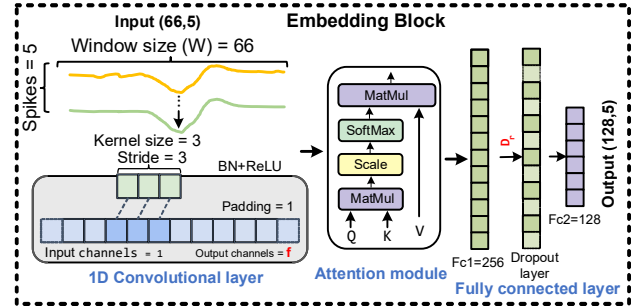


Fig.3. Details of embedding block of the FS-SS network. This module is used to standardize the input sequence, and also facilitates the use of few-shot capability in the deep learning processing pipeline. It includes a 1D convolutional layer with a variable number of output channels  $f$  depending on the size of the dataset, a batch normalization layer (BN), a self-attention mechanism unit and two fully connected layers (Fc1 and Fc2) outputs a 128-dimensional feature vector.

of parameters means less computation and lower computation cost which is more suited more to implantable brain processing. A batch normalization (BN) layer is connected after the convolution layer to speed up the training process [28].

The number of output channels  $f$  of the 1D convolutional layer shown in the embedding block in Fig.3 is calculated by linear interpolation according to the size of the  $N_d$ , where  $N_d$  represents the number of spikes used to train the model. Assuming that the maximum number of convolution kernels is  $f_{max}$ , the minimum number of convolution kernels is  $f_{min}$ , the maximum dataset size is  $N_{max}$ , and the minimum dataset size is  $N_{min}$ , the basic steps of the process are as follows:

a) Determine the normalization ratio  $P_{std}$  used for the dataset size:

$$P_{std} = \frac{N_d - N_{min}}{N_{max} - N_{min}} \quad (7)$$

b) Then, the initial value of the number of convolution kernels  $f_{linear}$  is calculated by linear interpolation:

$$f_{linear} = f_{min} + (P_{std} \times (f_{max} - f_{min})) \quad (8)$$

c) Finally,  $f_{linear}$  is rounded to the nearest power of 2 to get  $f$ , where  $\lceil \cdot \rceil$  represents rounding in Eq. (9). Adjusting the number of convolution kernels to a power of 2 can better utilize the memory alignment and parallel computing capabilities of the GPU:

$$f = 2^{\lceil \log_2(f_{linear}) + 0.5 \rceil} \quad (9)$$

The determination of  $f$  is performed in model initialization phase and is utilized in the embedding module during the training phase. A basic self-attention unit follows the convolutional layer, taking the output of the previous convolutional layer as input and further processing these features. Specifically, by adding the self-attention mechanism unit, the embedding module can generate the corresponding attention weight matrix by learning the primary time domain features obtained by convolution, so that the FS-SS model can focus more on the important time domain features of the spike signal. Finally, the output of the attention unit passes through two fully connected layers shown by Fc1 and Fc2 in Fig.3. The number of neurons in the first fully connected layer depends on the output feature dimensions of the attention unit after flattening, which is also closely related to the number of the kernels in the convolutional layer (because the input and output dimensions of the attention module are unchanged), which means that the



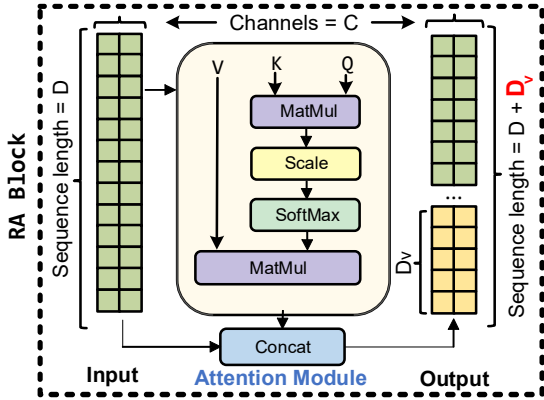


Fig.4. The RA block of the FS-SS network. The feature matrix of  $C \times D$  is linearly mapped using three fully connected layers to key (K), query (Q) and value (V). The length of processed input data D is mapped to  $D + D_v$  where  $D_v$  corresponds to the additional length introduced by the attention mechanism.

number of neurons in this layer is also adaptive. After the first fully connected layer (Fc1), a dropout layer is connected to prevent the "co-adaptation" problem [29]. The dropout rate is also determined by the size of the  $N_d$ , and linear interpolation is also used. In this work, the maximum dropout rate is 0.5 and the minimum dropout rate is 0.1. The dropout rate is inversely proportional to the size of the dataset, because small-scale datasets are prone to overfitting, so a larger dropout rate is required [29]. The output of the second fully connected layer (Fc2) is fixed to 128 to standardize the embedding results. This means that no matter what the scale of the input data is, the final embedding result length is always 128, thus ensuring the consistency of subsequent processing.

## 2) Residual Attention (RA) Block

Residual attention (RA) block based on the self-attention mechanism is considered in the network to capture the potential relationships of the features from the embedding module, and to provide more abstract sequence of features. The attention mechanism has been proven to be very effective in processing features with long-distance dependencies [27] so it is natural to apply the attention mechanism for tasks with multi-scale temporal features such as spike sorting shown in Fig.1.

The input of the RA accepts  $C$  channels with the input length of  $D$ , which corresponds to the number of extracted feature vectors from the previous stage. For example, the input of the first RA ( $C=128, D=5$ ) comes from the embedding module consisting of extracted features from 4 support spike waveforms and 1 query spike.

The number of channels processed by RA is  $C$ , so  $C$  will not change before and after passing through this module. The feature matrix of  $D \times C$  is linearly mapped using three fully connected layers to key (K), query (Q) and value (V). The corresponding fully connected layer sizes are recorded as  $D_Q$ ,  $D_K$  and  $D_V$ , and the corresponding weight matrices are called  $W_Q$ ,  $W_K$  and  $W_V$  respectively. The attention score is then calculated by the dot product between K and Q which reflects the dependencies between the features. Subsequently, in order to prevent the gradient saturation caused by excessive calculation results, the dot product result of K and Q is divided by the scaling factor  $\sqrt{D_K}$ . The Softmax function is then applied to convert the calculation result into a weight, and finally the

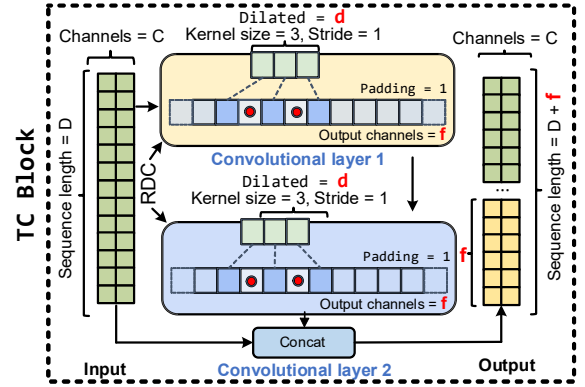


Fig.5. The RDC block used in the FS-SS network consists of two dilated convolutional layers (Layer 1 and Layer 2) and a residual operation, where the dilation rate (d) and the number of convolutional layer output channels are variable. Details of RDC are annotated on figure. TDC maps  $C \times D$  to an output feature vector of  $C \times (D+f)$ .

weight is applied to the value vector  $V$  to obtain the corresponding attention output. The whole process is expressed as:

$$Attention(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{D_K}}\right)V \quad (10)$$

Finally, in order to prevent degrading the performance of the model from due to the deepening of the network layers, residual connection is used in the RA block. The feature with a length of  $D$  becomes  $D_v$  after being processed by the attention mechanism (Eq. 10), and the original features and processed features are concatenated, so the final output feature length of the RA block is  $D + D_v$ .

## 3) Residual Dilated Convolutions (RDC) Block

Inspired by [26] and [30], a RDC block is designed that consists of two dilated convolutional layers and a residual operation as shown in Fig.5. The RDC block extracts more abstract features of the spike waveforms and it achieves a wider receptive field by sequentially stacking more RDC blocks.

Traditional convolution can be used to achieve a wider receptive field. However this needs to expand the size of the convolution kernels or stack more convolution layers [31]. Both methods inevitably introduce a sharp increase in computational complexity, limiting the practical use of deep neural networks in iBMIs. Therefore, dilation convolution is utilized in FS-SS to increase the receptive field of the convolution kernel without increasing the number of parameters (i.e. the implementation cost). This convolution method has significant advantages in processing multi-scale features and maintaining spatial resolution [32].

Dilated convolution significantly increases the receptive field without increasing the number of parameters [32] by scaling the convolution kernels as shown by red circles in Fig.5. Utilizing scaling factors, a spike waveform is decomposed to different time features. A temporal convolution (TC) module consists of two RDCs as shown in Fig.5. The TC modules are then stacked multiple times to achieve gradual increase of receptive field. Therefore, a smaller dilation rate is used to capture short-term peak features in the first RDC module. For instance, the first RDC has the dilation rate of 1, its corresponding convolution kernel size is set to 3 with a stride set to 1. Also, the padding in the first RDC is set to 1 to keep the input and output dimensions

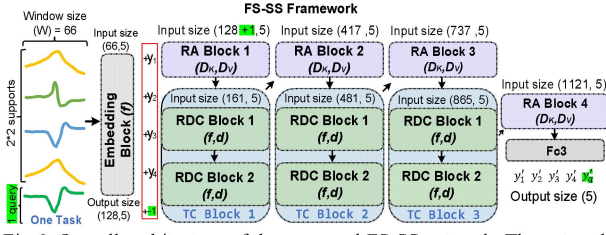


Fig. 6. Overall architecture of the proposed FS-SS network. The network consists of an embedding module, RA blocks and three TC modules. Each TC module stacks two RDC blocks. The figure shows the input and output dimensions of each module when the input spike data (66, 5). 66 is the length of the extracted spike waveform, and 5 represents 4 support set spikes and one query set spike. The dilation factors in the TDC units increase exponentially for effective expansion of the receptive field without introducing significant computation cost.

equal. For the subsequent RDC modules, larger dilation rates need to be used to obtain deeper receptive field. Following [32] and [33], the dilation rate of the RDC modules is set to twice that of the previous RDC module to capture the longer features of the spike signal. Therefore, the dilation rate corresponding to two sequentially connected RDC modules increases exponentially which can effectively expand the receptive field. The number of RDC modules ( $Z$ ) in a TC depends on the complexity of the meta-learning task, which is reflected here as the sum of the number of spikes in the support set and the query set  $S + Q$  determined by the following equation:

$$Z = \lceil \log_2(S + Q) \rceil \quad (11)$$

where  $\lceil \cdot \rceil$  represents rounding operation. As the RDC modules are stacked, the neural network will learn more and more abstract features. However, this will result in a serious gradient vanishing problem. As expressed in Eq. (12), the residual mechanism ensures that the gradient does not disappear during the backpropagation process by directly passing information to the subsequent layers [34]. For a input feature vector ( $x$ ) with a length of  $D$ , the dimension is changed to  $D + f$  after being processed by TC due to the residual operation, where  $f$  is determined by the number of convolution kernels. This allows to build and stack more RDC modules for implementing a deeper network.

$$RA \text{ output} = x + f(x, \theta) \quad (12)$$

### C. Architecture of FS-SS

As shown in Fig. 6, the entire FS-SS network consists of 1 embedding module, 4 RA blocks and 3 TC modules. Each TC module consists of 2 RDC blocks. Finally, a basic fully connected layer (Fc3) is added to predict the spike classes. Fig. 6 also shows the dimensional changes of the input samples after passing through each module under a task. In the proposed FS-SS network, 4 support spikes are included in the support set (i.e. 2-way 2-shot), and the query set consists of 1 spike highlighted in green. All 5 spikes are sent to the embedding module, and the final output feature vector dimension is standardized to 128 (128, 5). Subsequently, the true label of the support set is added to the 128-dimensional feature as an additional feature, and the label of the query set is blurred to a non-existent label -1 [26]. The concatenated 129-dimensional features are sent to the first RA block to learn the attention relationships between features. The attention results are concatenated with the residual path, and the final output dimension is 129+32 features where 32

represents the size of the  $D_V$  of the RA1 block. In the next three RA blocks (RA2, RA3 and RA4), due to the continuous increase in feature dimensions, the corresponding attention range also needs to be expanded accordingly. So the corresponding  $D_V$  of RA2, RA3 and RA4 modules also increases to 64, 128 and 256 respectively. In each RA block,  $D_K$  is equal to twice  $D_V$ , and  $D_Q$  is equal to  $D_K$ . As mentioned earlier, the TC module contains 2 RDC blocks, and the corresponding number of convolution kernels  $f$  is determined by the size of the dataset  $N_d$ . Similar to the RA block, the corresponding feature dimension increases by  $f$  after each TC module due to residual connections. Fig. 6 shows the input and output size of each module when  $f$  is 128.

During the network training phase, the learning rate was adjusted to 0.001, number of epochs was set to 50, the number of batches was set to 50, and the Adam optimizer and cross entropy loss function were used. The proportions of the training set, validation set and test set were 70%, 10% and 20% respectively.

## III. MATERIALS AND METHODS

### A. Datasets and Pre-processing

To compare the performance of the proposed method with other works, the Wave\_Clus spike bank was used [35]. The database in [35] comprises various average spike waveforms obtained from the neocortex and basal ganglia of humans. To replicate the background noise activity, attenuated spike waveforms selected at random from the data library were incorporated into the generated datasets. There are four datasets in the collected database, each has three spike mean waveforms and provides corresponding spike times and their labels. Besides, the four datasets are categorized according to the different degrees of difficulty (e.g., similarity of spike shape) and the noise levels. The datasets are labeled as C\_Easy1\_noise, C\_Easy2\_noise, C\_Difficult1\_noise, and C\_Difficult2\_noise, with noise levels represented by standard deviations ( $\sigma_N$ ) of 0.05, 0.1, 0.15, and 0.2. The terms "Easy" and "Difficult" refer to the similarity index between spike shapes in each dataset. Easy1 has also additional noise levels of 0.25, 0.3, 0.35 and 0.4 for further spike sorting performance analysis. Based on the time information for each spike, a sampling window consists of 66 samples was generated to extract the spike waveforms to examine the classification performance of the proposed FS-SS framework. Before the spike waveforms are fed to the network for classification, a simple min-max normalization is performed on the spike data.

### B. Evaluation Methods

Three metrics are used in this study to evaluate the sorting performance of the proposed framework, including precision, recall and accuracy. *Precision* is determined by dividing the number of true positive spikes (TPS) by the total number of all predicted positive spikes, including true positive spikes (TPS) and false positive spikes (FPS):

$$Precision = \frac{TPS}{TPS + FPS} \quad (13)$$

*Recall* is determined by dividing the number of true positive spikes (TPS) by the number of all actual positive spikes,

Table I Comparison of classification performance of the proposed FS-SS and other methods including Wave\_Clus, PCA-K and DSD under different datasets and noise levels. Precision, recall and the accuracy are calculated based on Eq. (13), (14) and (15) respectively.

Dataset	Noise levels	Proposed			Others (Accuracy)		
		Precision	Recall	Accuracy	Wave_Clus <sup>a</sup>	PCA-K <sup>b</sup>	DSD <sup>c</sup>
Easy1	0.05	99.18	99.16	99.16	98.82	<b>99.40</b>	99.13
Easy1	0.1	99.60	99.60	99.60	98.81	<b>99.65</b>	99.31
Easy1	0.15	99.64	99.64	<b>99.64</b>	98.72	99.45	99.13
Easy1	0.2	99.52	99.53	<b>99.53</b>	98.55	99.51	98.93
Easy2	0.05	99.60	99.60	<b>99.60</b>	96.88	95.21	97.31
Easy2	0.1	99.89	99.89	<b>99.89</b>	91.62	95.18	92.40
Easy2	0.15	99.53	99.52	<b>99.53</b>	91.30	96.77	92.23
Easy2	0.2	99.51	99.52	<b>99.51</b>	84.52	93.34	84.62
Difficult1	0.05	98.84	98.84	<b>98.84</b>	89.10	<b>98.84</b>	97.22
Difficult1	0.1	98.19	98.11	98.13	93.44	<b>98.93</b>	95.24
Difficult1	0.15	98.70	98.65	<b>98.66</b>	66.95	97.32	57.68
Difficult1	0.2	98.27	98.26	<b>98.26</b>	75.19	92.95	67.73
Difficult2	0.05	99.29	99.32	<b>99.30</b>	93.52	87.28	96.57
Difficult2	0.1	99.65	99.64	<b>99.65</b>	94.05	83.90	98.66
Difficult2	0.15	99.54	99.53	<b>99.54</b>	83.16	72.63	91.87
Difficult2	0.2	99.68	99.69	<b>99.69</b>	56.17	32.32	78.82
<b>Mean</b>		99.29	99.28	<b>99.28</b>	88.18	90.17	91.53

**a)** DWT(four-level Haar wavelet) and Kolmogorov–Smirnov used in Wave\_Clus [35].

**b)** PCA-K [36] uses three principle components in conjunction with K-means algorithm to classify the spikes.

**c)** Deep spike detection (DSD) [3] uses two CNNs to identify the active spike channels and to extract authentic spike data.

including true positive spikes (TPS) and false negative spikes (FNS) expressed as:

$$Recall = \frac{TPS}{TPS + FNS} \quad (14)$$

*Accuracy* indicates the proportion of correctly classified spikes, expressed as the sum of true positive (TPS) and true negative (TNS) divided by the number of all spikes, including true positive spikes (TPS), true negative spikes (TNS), false positive spikes (FPS) and false negative spikes (FNS), expressed as:

$$Accuracy = \frac{TPS + TNS}{TPS + FPS + FNS + TNS} \quad (15)$$

#### IV. EXPERIMENTS AND RESULTS

In this section, the proposed FS-SS framework is applied to the spike classification task. First, the performance of the FS-SS framework is compared with existing state-of-the-art algorithms. Then, to explore the adaptability of the model, explained dataset in Section III are divided into different portions to explore the performance of the FS-SS under different dataset sizes. Furthermore, attention visualization in various scenarios are presented in order to understand the feature extraction capability and interpretability of the proposed FS-SS network.

##### A. Evaluation of Classification Performance

The sorting performance of the proposed FS-SS framework and other state-of-the-arts including Wave\_Clus [35], combination of PCA and K-means (PCA-K) [36], and deep spike detection (DSD) [3] are shown in Table I. Algorithms in [3], [35] and [36] are briefly explained in the footnote of Table I. All three explained performance metrics in Section III. B, *Precision*, *Recall* and *Accuracy* are evaluated and reported in Table I for the considered spike sorting methods. To ensure the fairness of the comparisons, the mean sorting accuracy of seven epochs is calculated and two epochs with the highest and lowest performance values are removed (i.e. removing deviated

values). Therefore, the performance metrics listed in Table I are the average of 5 epochs.

The comparison of FS-SS, Wave\_Clus [35], PCA-K [36], and deep DSD [3] is performed across all spike shapes in “Easy” and “Difficult” datasets and the noise levels (0.05, 0.1, 0.15 and 0.2). It is observed from Table I that in Easy2\_0.05, the proposed FS-SS reaches *Precision* = 99.18% , a *Recall* of 99.16% and 99.16% *Accuracy*. This is lower than the PCA\_K method, but higher than the Wave\_Clus and DSD methods. When the noise standard deviation ( $\sigma_N$ ) is 0.1, the accuracy is still slightly lower than the PCA-K method. But when the noise level increases to 0.15 and 0.2, the accuracy of the proposed FS-SS is higher reaching to 99.64% and 99.53% respectively. In the other datasets, the FS-SS achieves higher sorting accuracy except the PCA\_K method on the Difficult1 dataset ( $\sigma_N=0.1$ ). This demonstrates the FS-SS network's ability to accurately identify the most informative features for sorting step, and its extreme robustness to spike similarity and noise level.

##### B. Learning with Different Dataset Proportions

As explained in Section II. B , linear interpolation was used to obtain optimal number of convolution kernels in feature extraction and to limit the size of the entire model to avoid unnecessary computations. Therefore, this section further evaluates the performance of scale-adaptive FS-SS model by using of 10 different data proportions ranging from 10% (or 0.1) to 100% (or 1). Data proportions refer to the number of the data samples used to simulate and verify the adaptation capability and classification performance of the FS-SS framework. Table II lists the total number of model parameters in different dataset proportions, as well as the corresponding hyperparameter information such as the number of convolution kernels and dropout rate. Fig.7 shows the classification performance of the FS-SS model for different training dataset proportions and different noise levels (taking the last 12 epochs and removing the maximum and minimum values).

From the overall trend in Fig.7, the FS-SS model shows a high classification accuracy in most cases, with slight fluctuations on

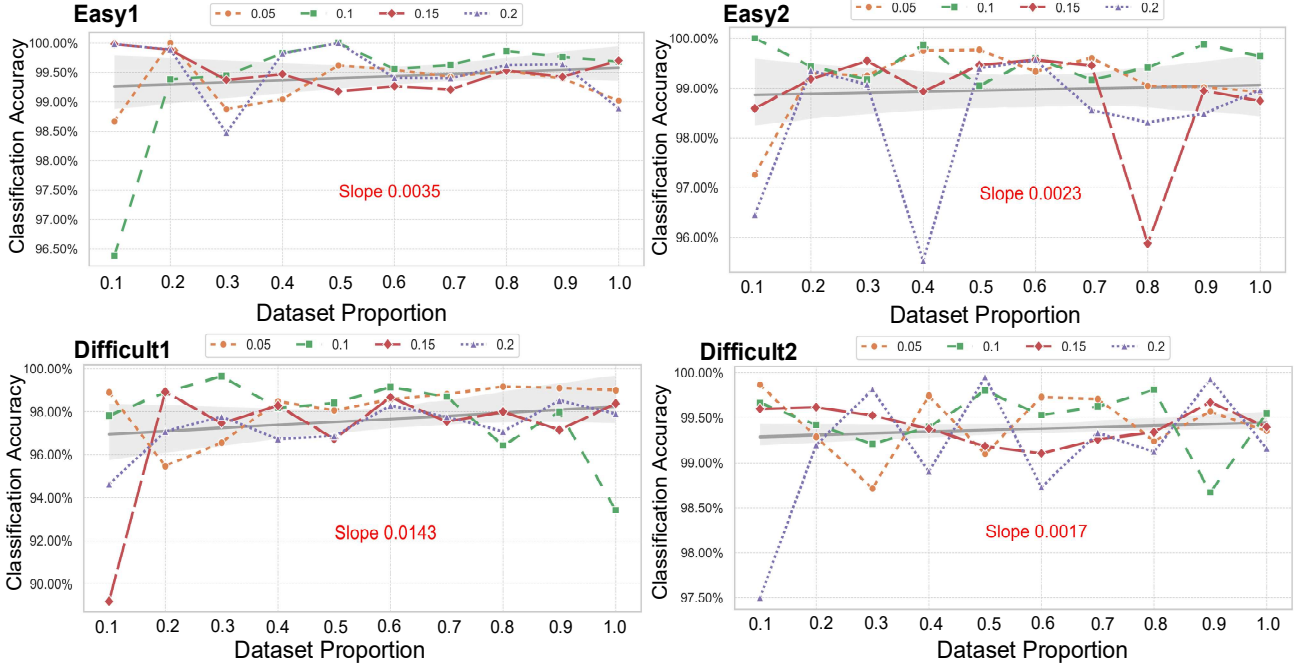


Fig. 7. Classification accuracy of Easy1, Easy2, Difficult1, and Difficult2 datasets at different standard deviations ( $\sigma_N$ ) of 0.01, 0.05, 0.15, and 0.2 versus dataset proportions ranging from 10% to 100%. The shaded area represents the mean accuracy that corresponds to more than >95% confidence level in the datasets. Slope in each figure is calculated from curve fitting (gray straight line), which indicates the impact of increasing the dataset proportion on the classification accuracy. Overall, FS-SS represents an outstanding robustness to the dataset proportion in the adaptation process.

small dataset proportions. As the dataset proportion continues to increase, the accuracy of the model tends to stabilize. Especially in the case of low noise ( $\sigma_N=0.05$  and  $\sigma_N=0.1$ ), the accuracy remains high across different data proportions showing extremely high robustness. In the case of high noise ( $\sigma_N=0.15$  and  $\sigma_N=0.2$ ), although the accuracy fluctuates to a certain extent, the overall mean accuracy remains high (i.e. minor fluctuations). This shows that the FS-SS has extremely fast adaptability to the recording channels changes over time even when 10% of data is used for training purposes. Assuming that there are originally 3514 spike waveforms for training and testing in Easy1 0.05, 10% of the spikes for training and testing results in 98.7% accuracy. The performance of the FS-SS network evolves quickly by employing meta-learning mechanism and adaptive hyperparameters even under small training data conditions.

A linear fit was performed to further understand the mean accuracy versus dataset proportions. The slope of the fit was used to quantify the impact of the dataset proportion on the model performance. The gray shaded areas shown in Fig. 7 represents an incremental trend and the calculated slopes are all positive which indicate the increase in the proportion of the datasets resulted in accuracy improvement in. For example, in Difficult1, the overall mean accuracy is dropped by 1.43% from 0.0143, when dataset proportion is reduced by 90%. This results in the total number of FS-SS parameters reduction from 3M to 0.9M as shown in Table II (e.g. when the number of convolution kernels is reduced from 64 to 8). In Easy1, Easy2 and Difficult2 datasets, reducing the dataset proportion by 90% only reduces the accuracy by 0.35%, 0.23%, and 0.17%.

This shows that the FS-SS model can successfully reduce the total number of parameters of the entire network by reducing

Table II Dropout rate, number of convolution kernels and model size for different data proportions.

Proportion	Dropout rate	Kernels numbers	Model Size
10% (0.1)	0.5	8	967,502
20% (0.2)	0.45	16	1,238,070
30% (0.3)	0.41	16	1,238,070
40% (0.4)	0.36	32	1,804,550
50% (0.5)	0.32	32	1,804,550
60% (0.6)	0.27	32	1,804,550
70% (0.7)	0.23	32	1,804,550
80% (0.8)	0.18	64	3,038,886
90% (0.9)	0.14	64	3,038,886
100% (1)	0.1	64	3,038,886

the number of convolution kernels when facing a small dataset, without decisively affecting the overall classification performance. This is associated to two reasons: first, dynamically adjusting the number of convolution kernels and the size of the dropout rate can effectively alleviate overfitting and improve the performance of the model on small datasets. Second, the FS-SS model utilizes meta-learning which is designed for adaptations using small datasets. By training on multiple related tasks, the FS-SS network can quickly learn key features when faced with new tasks. This proves the superior performance of the FS-SS network on spike classification with real-time adaptability.

### C. Spike Alignment and Feature Visualization

In order to further understand the feature extraction ability of the FS-SS, Fig. 8 shows two-dimensional (2D) scatter plots of the embedding module after t-SNE (t-Distributed Stochastic Neighbor Embedding) dimension reduction [37]. The left and right columns in Fig. 8 represent low ( $\sigma_N=0.05$ ) and high ( $\sigma_N=0.2$ ) noise datasets. Each row in Fig. 8 also represents the



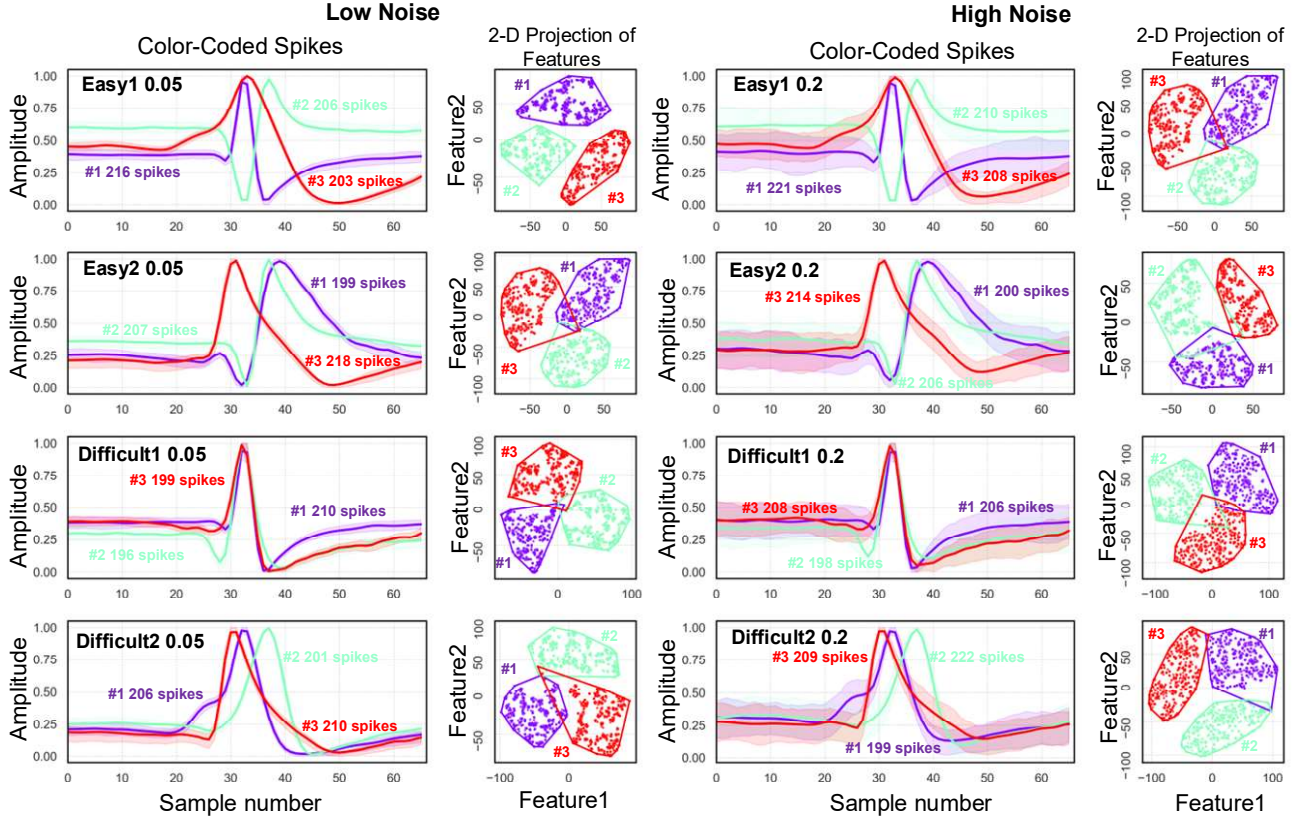


Fig.8. Alignment and feature visualization of spike waveforms of different datasets for  $\sigma_N=0.05$  (left column) and  $\sigma_N=0.2$  (right column). Under each column, color-coded spike waveforms are superimposed based on the ground-truth. The shaded part represents the 60% data distribution range. Annotated numbers next to the spikes show the number of predicted and aligned waveforms in each class. The scatter plot under left or right columns shows the visualization result of the 128-dimensional features after the embedding module using t-SNE dimensionality reduction. Each color represents a different class, and the envelope shows the boundary of the corresponding cluster.

type of dataset. The test spike waveforms are color-coded according to the ground-truth to better distinguish between the neurons. Aligned color-coded mean spike waveforms along with their shaded fluctuation range are shown for each dataset. It is observed from Fig.8 that, whether for the Easy dataset with lower similarity or the Difficult dataset with higher similarity, the mean waveforms are clear and identifiable which shows the proposed FS-SS network strong ability to extract time domain features of different spikes. Noisy conditions introduce a wider shaded area, however the superimposed classified waveforms still show an acceptable degree of separation. This shows that the noise has very little impact on the aligned spikes after classification, and the proposed FS-SS grasps the key features and maintains stable performance in the face of high noise. The 2D scatter plot on the right shows the t-SNE dimensionality reduction result of the embedding module after label prediction in the test set. It is observed that the feature points after dimensionality reduction show robust classification, and the boundaries between clusters corresponding to different classes are clear. Even in the case of high noise, there will be no high overlaps between different 2D cluster projection borders which shows that the FS-SS model can effectively encode features and so to distinguish features of different categories.

#### D. Visualization of Attention Results

This section illustrates the time domain feature heatmap of different spike waveforms in the embedding module since this

has a significant impact on the overall classification performance. Visualization of the attention distributions in different test datasets at a noise standard deviation of ( $\sigma_N=0.15$ ) are shown in Fig.9, for example the first row in Fig.9 corresponds to the Easy1\_0.15 dataset.

Fig.9 also superimposes the spike waveform, its first-order derivative (FD) by green dashed line and its second-order derivative (SD) using blue dotted line.  $FD_{max}$ ,  $FD_{min}$ ,  $SD_{max}$ ,  $SD_{min}$  representing maximum and minimum of FD and SD are marked in each figure for better understanding of attention heatmaps. It is observed from the top row in Fig. 9 that for the Easy1\_0.15, the attention heatmap mainly focuses on the  $FD_{max}$  and  $FD_{min}$  points of spike #1. There is also a part of attention distributed after  $SD_{max}$ . For spike #2, part of the attention is distributed in the rising phase near  $FD_{max}$  which represents the phase with the fastest voltage rise (i.e. maximum slope). Another part of the attention is distributed in the falling phase after  $SD_{min}$ , suggesting that the model considers this phase as the key morphological feature in spike #2 in Easy1\_0.15. For spike #3 in Easy1\_0.15, the attention distribution is slightly different, with one feature concentrated between  $SD_{min}$  and  $FD_{min}$  and the other part distributed after  $FD_{min}$ . This indicates that the proposed model believes the slow transition after  $FD_{min}$  is the key feature to distinguish spike #3 which is consistent with observations. These results strongly indicate that the proposed FS-SS network is not completely a "black box", on

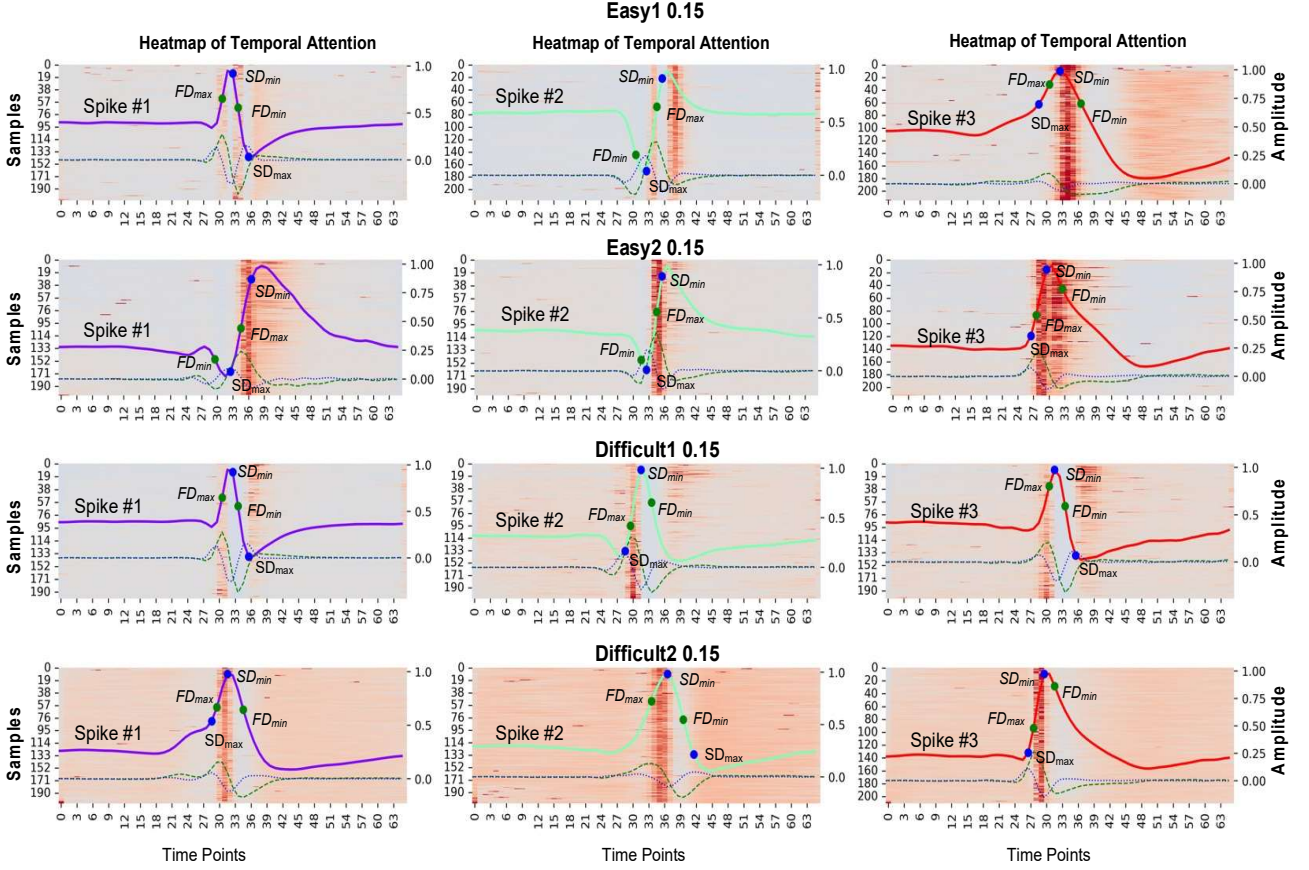


Fig.9. Attention heatmaps of three spike waveforms in different datasets in Easy1\_0.15 (first row), Easy2\_0.15 (second row), Difficult1\_0.15 (third row) and Difficult2\_0.15 (fourth row). The horizontal axis of each sub-graph represents the time point, and the vertical axis represents the sample number. The darker color reflects the degree of attention in the RA module to different time points. Three waveforms are also superimposed on the heatmaps: a) the mean of the spike waveform according to the predicted label, b) the green dotted line that represents its corresponding first-order derivative (FD), and the blue dotted line represents the second-order derivative (SD). The key features are marked including  $FD_{max}$  (maximum value of the first derivative),  $FD_{min}$  (minimum value of the first derivative),  $SD_{max}$  (maximum value of the second derivative), and  $SD_{min}$  (minimum value of the second derivative).

the contrary, it has a fairly strong electrophysiological interpretability.

In Easy2\_0.15 dataset, the distribution of attention is mainly concentrated on the rising phase of the spike aligned with  $FD_{max}$  and after the  $SD_{min}$ . In the Difficult1\_0.15 dataset with higher waveform similarity, there is no higher attention distribution near  $FD_{min}$  (i.e. the region with the fastest voltage drop) which indicates that the FS-SS believes this segment cannot effectively distinguish the three spike waveforms. More specifically, part of the attention is distributed around  $FD_{max}$ , where there is a fast voltage rise. Another part of the attention is distributed around  $SD_{max}$  and after that. This shows that the model observes more subtle differences in morphological features and places more attention on these two areas. The attention distribution corresponding to the Difficult2\_0.15 dataset is almost similar to Difficult1\_0.15, with more attention near  $FD_{max}$ . There is not so much attention distribution in the falling stage around  $FD_{min}$ . In short, the proposed model provides a fairly strong electrophysiological interpretation through attention in distinguishing the most informative features in different spikes, and can dynamically adjust attention according to the characteristics of different morphologies. The dynamic attention distribution adjustment is not only used to assist in observing direct or indirect features,

but also demonstrates the superior performance of the FS-SS network in spike sorting.

## V. CONCLUSION

This paper proposed a FS-SS framework based on meta-learning and attention mechanism which is suitable for implantable brain processing (referred as spike sorting), where collecting training data is an extremely challenging task. The proposed framework consists of crucial building blocks including an embedding module, residual attention (RA) module and residual dilated convolution (RDC) module. Combination of these key modules ensure quick generalization of learning with only two-shot from each class by building upon the previous experiences in the learning phase, and also extraction of the most informative features in the classification stage. The results show that the proposed framework achieves leading performance on the public dataset. Specifically, when using all samples, FS-SS achieved an average precision of 99.29%, an average recall of 99.28%, and an average classification accuracy of 99.28%. Even on sub-datasets with high noise or high similarity, the proposed framework shows robust outcomes. In addition, by designing hyperparameters that can be dynamically adjusted according to the number of

training data, the parameter size of the model is adaptive. When the number of samples is reduced by 90%, the model size becomes 31.8% of the original, while the average accuracy only decreases by 0.55%. This makes it highly suitable for implantable on-chip spike sorting algorithms with extremely high performance and power requirements. Finally, the t-SNE results of the embedding layer and the visualization of the time-domain attention results prove that the framework has a certain physiological explanation rationality and provides more trust in terms of sorting robustness. In future work, the FS-SS will be further improved and actual tests will be conducted using in-vivo datasets.

## REFERENCES

- [1] M. Zamani, and A. Demosthenous, "Feature extraction using extrema sampling of discrete derivatives for spike sorting in implantable upper-limb neural prostheses," *IEEE Trans Neural Syst Rehabil Eng*, vol. 22, no. 4, pp. 716-26, Jul, 2014.
- [2] M. Zamani, D. Jiang, and A. Demosthenous, "An Adaptive Neural Spike Processor With Embedded Active Learning for Improved Unsupervised Sorting Accuracy," *IEEE Trans Biomed Circuits Syst*, vol. 12, no. 3, pp. 665-676, Jun, 2018.
- [3] C. O. Okreghe, M. Zamani, and A. Demosthenous, "A deep neural network-based spike sorting with improved channel selection and artefact removal," *IEEE Access*, vol. 11, pp. 15131-15143, 2023.
- [4] M. Zamani, J. Sokolić, D. Jiang, F. Renna, M. R. Rodrigues, and A. Demosthenous, "Accurate, very low computational complexity spike sorting using unsupervised matched subspace learning," *IEEE Transactions on Biomedical Circuits and Systems*, vol. 14, no. 2, pp. 221-231, 2020.
- [5] L. M. Meyer, M. Zamani, and A. Demosthenous, "Binarized Neural Networks for Resource-Efficient Spike Sorting," *IEEE Access*, doi: 10.1109/ACCESS.2025.3552008, 2025.
- [6] M. Zamani, C. Okreghe, and A. Demosthenous, "Efficient approximation of action potentials with high-order shape preservation in unsupervised spike sorting," in Proc. 44th Annu. Int. Conf. IEEE Eng. Med. Biol. Soc. (EMBC), Jul. 2022, pp. 4884-4887.
- [7] A. Mohammed, M. Zamani, R. Bayford, and A. Demosthenous, "Toward on-demand deep brain stimulation using online Parkinson's disease prediction driven by dynamic detection," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 25, no. 12, pp. 2441-2452, 2017.
- [8] T. S. Davis, H. A. Wark, D. Hutchinson, D. J. Warren, K. O'Neill, T. Scheinblum, G. A. Clark, R. A. Normann, and B. Greger, "Restoring motor control and sensory feedback in people with upper extremity amputations using arrays of 96 microelectrodes implanted in the median and ulnar nerves," *Journal of neural engineering*, vol. 13, no. 3, pp. 036001, 2016.
- [9] M. Capogrosso, T. Milekovic, D. Borton, F. Wagner, E. M. Morand, J.-B. Mignardot, N. Buse, J. Gandar, Q. Barraud, and D. Xing, "A brain-spine interface alleviating gait deficits after spinal cord injury in primates," *Nature*, vol. 539, no. 7628, pp. 284-288, 2016.
- [10] N. A. Steinmetz, C. Aydin, A. Lebedeva, M. Okun, M. Pachitariu, M. Bauza, M. Beau, J. Bhagat, C. Böhm, and M. Broux, "Neuropixels 2.0: A miniaturized high-density probe for stable, long-term brain recordings," *Science*, vol. 372, no. 6539, pp. eabf4588, 2021.
- [11] A. Schüz, and G. Palm, "Density of neurons and synapses in the cerebral cortex of the mouse," *Journal of Comparative Neurology*, vol. 286, no. 4, pp. 442-455, 1989.
- [12] S. C. Colachis, C. F. Dunlap, N. V. Annetta, S. M. Tamrakar, M. A. Bockbrader, and D. A. Friedenberg, "Long-term intracortical microelectrode array performance in a human: a 5 year retrospective analysis," *Journal of Neural Engineering*, vol. 18, no. 4, pp. 0460d7, 2021.
- [13] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, pp. 436-444, 2015.
- [14] L. Meyer, M. Zamani, J. Rokai, and A. Demosthenous, "Deep learning-based spike sorting: a survey," *Journal of Neural Engineering*, doi: 10.1088/1741-2552/ad8b6c, 2024.
- [15] T. Wu, A. Rátai, K. Schlett, L. Grand, and Z. Yang, "Learning to sort: few-shot spike sorting with adversarial representation learning," pp. 713-716.
- [16] E. Rahimian, S. Zabihi, A. Asif, D. Farina, S. F. Atashzar, and A. Mohammadi, "FS-HGR: Few-shot learning for hand gesture recognition via electromyography," *IEEE transactions on neural systems and rehabilitation engineering*, vol. 29, pp. 1004-1015, 2021.
- [17] M. Atzori, A. Gijssberts, S. Heynen, A.-G. M. Hager, O. Deriaz, P. Van Der Smagt, C. Castellini, B. Caputo, and H. Müller, "Building the Ninapro database: A resource for the biorobotics community," pp. 1258-1265.
- [18] K. Pałczyński, S. Śmigiel, D. Ledziński, and S. Bujnowski, "Study of the few-shot learning for ECG classification based on the PTB-XL dataset," *Sensors*, vol. 22, no. 3, pp. 904, 2022.
- [19] P. Wagner, N. Strothoff, R.-D. Bousseljot, D. Kreiseler, F. I. Lunze, W. Samek, and T. Schaeffter, "PTB-XL, a large publicly available electrocardiography dataset," *Scientific data*, vol. 7, no. 1, pp. 1-15, 2020.
- [20] T. Hospedales, A. Antoniou, P. Micaelli, and A. Storkey, "Meta-Learning in Neural Networks: A Survey," *IEEE Trans Pattern Anal Mach Intell*, vol. 44, no. 9, pp. 5149-5169, Sep, 2022.
- [21] Y. Chen, Z. Liu, H. Xu, T. Darrell, and X. Wang, "Meta-baseline: Exploring simple meta-learning for few-shot learning," pp. 9062-9071.
- [22] J. Li, F. Wang, H. Huang, F. Qi, and J. Pan, "A novel semi-supervised meta learning method for subject-transfer brain-computer interface," *Neural Networks*, vol. 163, pp. 195-204, 2023.
- [23] T. Duan, M. A. Shaikh, M. Chauhan, J. Chu, R. K. Srihari, A. Pathak, and S. N. Srihar, "Meta learn on constrained transfer learning for low resource cross subject EEG classification," *IEEE Access*, vol. 8, pp. 224791-224802, 2020.
- [24] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," pp. 1126-1135.
- [25] O. Vinyals, C. Blundell, T. Lillicrap, and D. Wierstra, "Matching networks for one shot learning," *Advances in neural information processing systems*, vol. 29, 2016.
- [26] N. Mishra, M. Rohaninejad, X. Chen, and P. Abbeel, "A simple neural attentive meta-learner," *arXiv preprint arXiv:1707.03141*, 2017.
- [27] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.
- [28] S. Ioffe, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *arXiv preprint arXiv:1502.03167*, 2015.
- [29] G. Hinton, "Improving neural networks by preventing co-adaptation of feature detectors," *arXiv preprint arXiv:1207.0580*, 2012.
- [30] S. Bai, J. Z. Kolter, and V. Koltun, "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling," *arXiv preprint arXiv:1803.01271*, 2018.
- [31] K. Simonyan, and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.
- [32] F. Yu, and V. Koltun, "Multi-scale context aggregation by dilated convolutions," *arXiv preprint arXiv:1511.07122*, 2015.
- [33] A. Van Den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, "Wavenet: A generative model for raw audio," *arXiv preprint arXiv:1609.03499*, vol. 12, 2016.
- [34] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," pp. 770-778.
- [35] R. Q. Quiroga, Z. Nadasdy, and Y. Ben-Shaul, "Unsupervised spike detection and sorting with wavelets and superparamagnetic clustering," *Neural Comput*, vol. 16, no. 8, pp. 1661-87, Aug, 2004.
- [36] I. Y. Park, J. Eom, H. Jang, S. Kim, S. Park, Y. Huh, and D. Hwang, "Deep learning-based template matching spike classification for extracellular recordings," *Applied Sciences*, vol. 10, no. 1, pp. 301, 2019.
- [37] L. Van der Maaten, and G. Hinton, "Visualizing data using t-SNE," *Journal of machine learning research*, vol. 9, no. 11, 2008.